LA RIVISTA DI ELETTRONICA APPLICATA, TECNOLOGIE E PRODOTTI

FareELETTRONICA

N° 234 - DICEMBRE 2004 - ANNO 20

€ 4,50 - Frs 9,00

TUTTO... MA PROPRIO TUTTO SUGLI ALIMENTATORI SWITCHING

FREQUENZIMETRO DIGITALE BF 4,5 DIGIT LED

FEBOT
Il firmware della scheda madre

SMARTCARD
Applicazioni con le SIM

Poste Italiane Spa - Spedizione in abbonamento Postale - D.L. 353/2003 (conv. In L. 27/02/2004 n. 46) art. 1, comma1, DCB Milano.
In caso di mancato recapito, restituire all'editore che si impegna a pagare la relativa tassa presso il CPM di Roserio - Milano

ISSN 1591-2272
40234
9 771591 227008

INWARE EDIZIONI

editoriale

EVOLUZIONE

Lo stesso giorno dell'acquisto di Fare Elettronica da parte della attuale casa editrice (Inware Edizioni) avvenuto nell'ottobre 2003, ci siamo riuniti per tenere il primo degli incontro mensili che oggi chiamiamo "*evolution meeting*".
Ci siamo ritrovati nella sala riunioni e, dopo una lunga serie di interventi, piani, decine di fogli riempiti con idee e proposte, siamo stati tutti concordi nell'affermare che da quel giorno in poi il nostro motto sarebbe stato "evoluzione".
Mai fermarsi, mai riposarsi sugli allori, ma lavorare continuamente per arrivare ad un risultato sempre migliore e subito dopo porsi un nuovo obiettivo, sempre più ambizioso, alla continua ricerca della "rivista perfetta", della rivista che ognuno di noi ha in mente e che voi, cari lettori, state delineando con i vostri apprezzamenti e consigli.
Una rivista moderna, ricca di contenuti innovativi, che tratti argomenti volti alla formazione dei lettori, non solo sterili progetti "realizza e getta", una rivista che fosse piacevole da leggere ed approfondire, non soltanto da sfogliare distrattamente, da collezionare non solo perché "tanto ho tutti i numeri" ma da conservare come uno strumento di laboratorio, sempre pronto ad aiutare nella soluzione di un problema, uno strumento che può tornare utile in qualsiasi momento.
Ad oggi, stando ai vostri apprezzamenti, siamo sulla buona strada; gli obiettivi sono sempre nuovi e stimolanti, i risultati davvero incoraggianti. È un processo lungo e che richiede molto impegno, ma abbiamo fatto molta strada ed altrettanta ne faremo.
Questo si chiama *evoluzione*: "*uno sviluppo lento e graduale; svolgimento da una forma a un'altra, generalmente più completa e perfetta*" (fonte Garzanti).
Lo avete vissuto in prima persona, tutte le iniziative che hanno accompagnato la rivista in questo ultimo anno, tutti i miglioramenti, il nuovo sito, la partecipazione della redazione alle fiere… tutti piccoli passi nel nostro percorso evolutivo.
Non è stato facile e non lo sarà neanche quello che ci prefiggiamo per il prossimo anno, un progetto ancora più ambizioso, ma a noi piacciono le sfide ed amiamo il nostro lavoro, per questo vi consiglio di non perdervi il numero di Gennaio 2005 sarà una pietra miliare nella storia della nostra amata rivista.

Prima di introdurre il numero che state per leggere lasciate che vi presenti il nostro nuovo Direttore Responsabile: Antonio Cirella.
Molti lettori hanno già avuto la possibilità di conoscerlo personalmente via email o tramite qualche articolo scritto in passato. Nel prossimo anno avrà un suo spazio in questo editoriale, parlando di argomenti di attualità, mentre io continuerò a parlarvi dei contenuti della rivista.
Noi tutti ringraziamo Giancarmelo Moroni per il preziosissimo contributo fin qui dato alla rivista ed auguriamo un futuro ricco di successi al nuovo Direttore.

Non mi rimane molto spazio per descrivere il bellissimo numero che state per leggere, ma voglio segnalarvi la prima parte del nuovo corso sugli *alimentatori switching*. Questo è un argomento sul quale Fare Elettronica è all'avanguardia, come nostro solito non ci siamo limitati a un semplice articolo o progetto, ma siamo andati oltre con un corso completo che vi accompagnerà per tutto il 2005, tanta teoria spiegata in parole semplici ma anche tanta pratica, tanti progetti completi di alimentatori switching di diversa tipologia. Se a questo tutorial aggiungete il nuovo corso di elettronica *Elettronicando* che partirà con il numero di Gennaio 2005, avrete soltanto una minima idea delle novità che vi aspettano il prossimo anno.

Bene, concludo questo mio intervento augurandovi, da parte mia e di tutta la redazione, un felice Natale ed un ricchissimo anno nuovo.
Vi rinnovo l'appuntamento in edicola a Gennaio 2005 per un nuovo ed entusiasmante numero di Fare Elettronica.

Tiziano Galizia
t.galizia@farelettronica.com

www.farelettronica.com

DIRETTORE RESPONSABILE:
Antonio Cirella (a.cirella@farelettronica.com)

DIRETTORE DI REDAZIONE:
Tiziano Galizia (t.galizia@farelettronica.com)

PROGETTO GRAFICO E IMPAGINAZIONE:
Graficonsult - Milano (info@graficonsult.com)

HANNO COLLABORATO:
Maurizio Del Corso, Romano Bernarducci, Umberto Fabris,
Massimiliano Bracci, Andrea Perilli, Sandro Romagnoli, Raffaello Bonghi,
Gianroberto Negri, Antonio Di Stefano, Nico Grilloni, Giuseppe Modugno,

DIREZIONE - REDAZIONE - PUBBLICITÁ
INWARE srl
Via Cadorna, 27/31 - 20032 Cormano (MI)
Tel. 02.66504794 - 02.66504755 - Fax 02.66508225
info@inware.it - www.inwaredizioni.it

STAMPA:
ROTO 2000
Via Leonardo da Vinci, 18/20 - 20080 Casarile (MI)

DISTRIBUZIONE:
Parrini & C. S.p.a.
Viale Forlanini, 23 - 20134 Milano.
Il periodico Fare Elettronica è in attesa del numero di iscrizione al ROC

UFFICIO ABBONAMENTI
PARRINI & C. S.p.a. Servizio abbonamenti
Viale Forlanini, 23 - 20134 Milano
Per informazioni, sottoscrizione o rinnovo dell'abbonamento:
Telefono: 02.66504794 - Fax: 02.66508225
Email: abbonamenti@farelettronica.com

Poste Italiane Spa - Spedizione in abbonamento Postale - D.L. 353/2003
(conv. In L. 27/02/2004 n. 46) art. 1, comma1, DCB Milano
Abbonamento per l'Italia: € 39,00
Abbonamento per l'estero: € 99,00
Per la sottoscrizione degli abbonamenti, utilizzare il modulo stampato in ultima pagina.
Gli arretrati potranno essere richiesti, per iscritto, al seguente costo:
Numero singolo: € 7,50
Numero con allegato: € 8,50
Numero doppio: € 9,00

Autorizzazione alla pubblicazione del Tribunale di Milano n. 647 del 17/11/2003 INWARE srl.

Richieste di assistenza

Per richiedere assistenza o chiarimenti sugli articoli pubblicati, vi preghiamo di contattare l'autore, il cui nome ed indirizzo email è sempre riportato sotto il titolo dell'articolo stesso.
Nel caso ciò non fosse possibile potete scrivere a mailbox@farelettronica.com, ricordandovi di specificare il numero della rivista ed il titolo dell'articolo per il quale chiedete chiarimenti, oltre al vostro nome, cognome ed indirizzo email.
Tutte le richieste con informazioni insufficienti o anonime non saranno prese in considerazione.

Collaborare con Fare Elettronica

La redazione di Fare Elettronica è alla ricerca di collaboratori per la stesura di articoli, progetti, tutorials, rubriche e libri.
Le richieste di collaborazione vanno indirizzate a Tiziano Galizia (t.galizia@farelettronica.com) e accompagnate, se possibile, da una breve descrizione delle vostre competenze tecniche e/o editoriali, oltre che da un elenco degli argomenti e/o progetti che desiderate proporre.

Come contattarci

Indirizzo email della Redazione:
redazione@farelettronica.com
Indirizzo email dell'Ufficio Abbonamenti:
abbonamenti@farelettronica.com
I nostri numeri telefonici:
Telefono 02.66504794 Fax 02.66508225
Il nostro indirizzo postale:
INWARE Edizioni
Via Cadorna, 27/31
20032 Cormano (MI)

Elenco inserzionisti

SOMMARIO

hardware

tecnologie sperimentali

praticamente

tutorial

Mhz

robomania

rubriche

Dubbi, perplessità, malfunzionamenti, opinioni, commenti o richieste?

Inviateli a: ***mailbox@farelettronica.com***

Oppure scriveta a:
Mailbox - Redazione di Fare Elettronica
Inware srl
Via Cadorna, 27/31 - 20032 Cormano (MI)

Le lettere più interessanti saranno pubblicate in queste pagine. Per quanto possibile, inoltre, cercheremo di dare una risposta privata a chiunque ci scriverà via email.

D ALIMENTATORE

Spett. Redazione, vorrei realizzare un alimentatore che fornisca in uscita 24V-8A. Volevo sapere se potevo usare i circuiti descritti dal signor Grilloni oppure delle varianti.

Daniele Rondi

R Il circuito richiesto è riportato in figura 1. Si

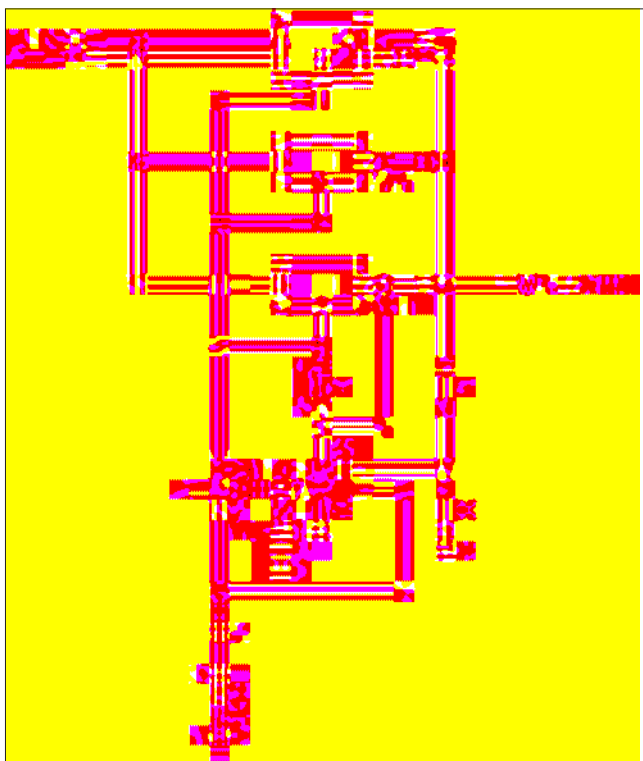



Figura 1: *Alimentatore 5-25V 10 A*

tratta di un alimentatore in grado di fornire in uscita una tensione variabile da circa 5 V a oltre 25 V per una corrente di carico di 10 A. L'integrato utilizzato è un LM350 che, derivato dal più noto LM117, è in grado di erogare oltre 3 A.
Qui ne sono stati messi in parallelo tre compensando, con le tre resistenze da 0,1 Ω messe in serie a ciascuna uscita, eventuali differenze fra l'uno e l'altro.
Si suggerisce di utilizzare gli LM350 in involucro TO-3 (e non TO-220) e di disporli su opportuno radiatore.

D POTENZE AUDIO

Spettabile Redazione di Fare Elettronica, vorrei conoscere il significato della potenza RMS, potenza musicale e potenza picco-picco di un amplificatore audio.

Giorgio Mariani

R La potenza RMS (P_{RMS}) è la potenza massima che l'amplificatore è in grado di erogare sul carico (l'altoparlante). Questa viene calcolata supponendo di trasmettere un segnale sinusoidale di ampiezza pari alla tensione di alimentazione. È ovvio che è una potenza "fittizia" in quanto, in queste condizioni, la distorsione del segnale assumerebbe valori inammissibili. La potenza musicale (P_{mus}) e quella picco-picco (P_{pp}) sono legate alla

P_{RMS} dalla relazione:

$$P_{pp}=2*P_{mus}=4*P_{RMS}$$

La potenza musicale viene calcolata supponendo di trasmettere un'onda quadra della stessa ampiezza della sinusoide impiegata per il calcolo della P_{RMS}.
Se si confrontano diversi amplificatori è necessario comparare lo stesso tipo di potenze.
Nella scelta di un amplificatore audio, la potenza non è il solo parametro importante. Esistono infatti altre grandezze fondamentali che identificano la qualità di un amplificatore, come ad esempio il THD (Total Harmonic Distorsion) e la banda a -3dB. La banda a -3dB è il campo di frequenze in cui il segnale viene attenuato di un fattore inferiore a 2 rispetto al punto di massimo guadagno. La banda a -3dB dovrebbe comprendere tutta la soglia dell'udibile (20Hz-20KHz).
Il THD è invece un indice di quanto il segnale audio viene distorto dall'amplificatore. Tanto più è basso il THD tanto migliore è l'amplificatore. Valori accettabili devono essere inferiori all'1% per una data potenza. Se ad esempio si legge che un amplificatore ha una potenza di 120W RMS e THD pari allo 0,5% a 40W, significa che l'amplificatore può erogare una potenza massima di 120W, ma che la migliore resa sonora si ha nell'intorno dei 40W.

D TRASFORMATORI

Mi sto appassionando all'elettronica grazie anche alla vostra rivista.
Potreste dedicare, se già non lo avete fatto, qualche articolo sul funzionamento dei trasformatori, riportando magari qualche esempio con gli andamenti dei segnali al primario ed al secondario.
Sul numero di Luglio/Agosto 2004, nella rubrica NOTEPAD, avete presentato un inverter 40W dove è presente un trasformatore dove ai morsetti del primario sono connessi due transistors. Potreste spiegarmi il funzionamento?

Fabio Giardina

R Dedicheremo la puntata della rubrica "Praticamente" di Gennaio 2005 per approfondire il funzionamento di un trasformatore. Rispondiamo invece alla domanda relativa ai transistors presenti nel circuito al quale fa riferimento il lettore e che, per comodità, è riportato in figura 2.
Un inverter è un circuito in grado di fornire la tensione 220Vac a partire da una tensione continua. Per fare questo si sfrutta un trasformatore (usato al contrario, quindi alimentato dal secondario) e si sfrutta un teorema matematico (teorema di Fourier) secondo il quale qualsiasi segnale periodico, può essere scomposto in tanti segnali sinusoidali (armoniche) di frequenza pari e multipla del segnale di partenza. L'astuzia consiste dunque nel pilotare il trasformatore con un'onda quadra a 50Hz. Le sue armoniche saranno onde sinusoidali di frequenza 50Hz (fondamentale), 100Hz (prima armonica), 150Hz (seconda armonica), 200Hz, etc.. (in realtà un'onda quadra ha solo armoniche dispari quindi 50Hz, 150Hz, 250Hz, ecc..). Tutte le armoniche superiori (dalla prima in poi) vengono attenuate fortemente dall'induttanza degli avvolgimenti del trasformatore, quindi è come se il trasformatore fosse alimentato solo dalla fondamentale, ovvero un'onda sinusoidale a 50Hz. Al secondario avremo quindi i 220Vac.
In che modo si ottiene l'onda quadra a 50Hz per alimentare il trasformatore? Ecco che entrano in gioco i due transistori. Si usa un trasformatore a presa centrale alla quale si connette il positivo di alimentazione (ad es. 12Vcc), mentre agli altri due morsetti si collegano i transistori (come mostrato in figura). Questi funzionano come degli interruttori comandati che vengono aperti e chiusi alternativamente da un circuito di controllo (il CD4047 nel caso dell'inverter in figura) in modo da applicare un'onda quadra a 50Hz all'ingresso del trasformatore.

Figura 2: *Inverter 40W*

NEWS

424

*Questo spazio è gentilmente offerto da **EONews**, il Quindicinale di notizie e commenti per l'industria elettronica di **VNU Business Publications Italia**.*

IDEE DI PROGETTO: LA NUOVA INIZIATIVA SUL WEB DI ELETTRONICA OGGI

"Idee di progetto – Design Ideas" è la nuova rubrica di Elettronica Oggi che è stata lanciata dal mese di Settembre 2003. Caratteristica saliente di questa nuova iniziativa è che sarà completamente ed esclusivamente on line ed accessibile dal sito www.ilb2b.it.
L'obbiettivo principale è creare una vera e propria libreria di idee alla quale tutti coloro che operano in maniera professionale nel mondo dell'elettronica possano "catturare" informazioni e suggerimenti utili per il loro lavoro quotidiano. Questo nuovo strumento vi permette di scaricare, con un solo click, tutte le risorse necessarie per risolvere velocemente qualsiasi problema e, in ultima analisi, minimizzare il time to market.

INDUTTORI DI POTENZA

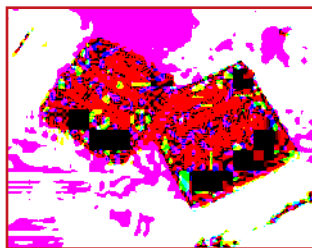
CoEv Magnetics (Atm electronics) ha annunciato una nuova serie Dzm di induttori di potenza senza piombo, a bassa tensione e ad alta corrente, progettati per applicazioni di alimentatori switching e componenti elettronici portatili.
Al contrario di altri induttori a basso costo gli induttori in questione non risentono del deterioramento determinato da effetti termici tipico delle solite polveri di ferro e permettono di immagazzinare una maggiore energia, sono senza piombo e compatibili con l'Rohs.
Il basso profilo e le migliori performance dalla Dc di bias consentono ulteriori possibilità d'impiego e di incrementare le prestazioni dei prodotti senza sacrificare spazio esterno utile.

readerservice.it 424 - 51

BOOST CONVERTER

Lt1935 di Linear Technology (Silverstar-Celdis) è un regolatore switching Sot-23 della massima potenza industriale. Si tratta di un convertitore Dc/Dc step-up da 1,2 MHz, 40 V, 2 A racchiuso in un package ThinSot. L'ampio intervallo delle tensioni di ingresso (da 2,3 a 16 V) gli consente di funzionare da batterie a ioni di litio a celle singole fino a rail di ingresso 15 V fisso e di erogare uscite fino a 38 V. La frequenza di commutazione di 1,2 MHz costante permette di mantenere il rumore fuori dai circuiti sensibili al rumore e l'uso di minuscoli condensatori e induttori. È perfetto per applicazioni vincolate da problemi di spazio.

readerservice.it 424 - 53

CHIP DI TEMPORIZZAZIONE

Zarlink Semiconductor (Unique Memec) ha annunciato una coppia di chip di temporizzazione che in combinazione presentano un set di caratteristiche complete ed ottime prestazioni per sistemi Sonet/Sdh e Pdh e si finalizzano a schede di linea utilizzate in una vasta gamma di apparecchiature da enterprise a network core. Oltre a prestazioni di tremolio di 20 psrms, Zl30106 (Dpll digitale) può sincronizzare riferimenti di ingressi primari e secondari a coppie di impulsi di clock e sincronizzazione, mentre Zl30416 (Pll analogico) è specificatamente progettato per soddisfare i requisiti di prestazioni e caratteristiche di applicazioni di schede di linea da velocità di trasmissione Oc-3/Stm-1 a Oc-192/Stm-64.

readerservice.it 424 - 54

SOLUZIONE GPS

La soluzione Gps Antaris Tim-Lf di u-blox (Power Source) è stata scelta da Navsys e Geo++ per i loro sistemi di posizionamento relativo e lettura di assetti ad alta precisione. Fornisce un'accuratezza di +/-1 cm in posizionamenti e 0.2 gradi in rotte, ruoli e passi con una velocità di aggiornamento fino a 10 Hz. Il sistema di posizionamento e lettura assetti 3D Smartpos Navsys utilizza il sensore Q16 Smartgps contenente quattro ricevitori Gps Tim-Lf integrati in un piccolo involucro misurante 11 x 11 x 3,5 cm che è connesso ad una matrice di quattro antenne.
I dati grezzi (codici e portanti) del sensore vengono elaborati dalla soluzione di reti Rt a multistazione Geo++ per ottenere una soluzione comune a tutti e quattro gli ingressi dei ricevitori.

readerservice.it 424 - 55

CONNETTORI DA PANNELLO

Compel Electronics ha sviluppato un'innovativa versione di connettori

7/16 da pannello, che si aggiunge a quelle già esistenti e largamente utilizzate per applicazioni Radio Mobile e Broadcasting. Questa versione è stata progettata come un blocco compatto, con le parti di accoppiamento in metallo e con la parte di fissaggio in plastica, disponibile in vari colori. Il contatto esterno può essere scelto, come prima, sia nella versione elastica che rigida. Tutte le caratteristiche elettriche, meccaniche e ambientali rimangono assolutamente le stesse della versione in metallo.

readerservice.it 424 - 56

CONDENSATORI RFI

Wima (Sge-Syscom) offre ora condensatori Rfi con dielettrico polipropilene in aggiunta al metallizzato. Le serie Mkp-X2 e Mkp-Y2 sono disponibili con valori da 1000 pF a 2,2 microF e tensioni di 275 e 300 Vac rispettivamente per le serie X2 e Y2. Hanno inoltre approvazioni Enec/Vde in accordo a Din En 132 40/Iec 60384-14/2 e Ul 1414/Cas C 22.2 N.1 (cUlus). Hanno caratteristiche di alta capacità e piccole dimensioni accompagnate da un prezzo competitivo. Con queste serie viene offerta un'ampia linea di condensatori Wima serie Rfi che include sia quelli in versione leaded con dielettrico polipropilene e carta sia quelli in Smd con dielettrico e carta.

readerservice.it 424 - 59

TRASDUTTORI DI CORRENTE

Le serie di trasduttori di corrente Hop e Htr di Lem, basate sul principio della tecnologia ad effetto Hall Open Loop, permettono misure di corrente Ac, Dc e impulsive da 50 a 2000 Arms. Prestazioni elevate, dimensioni compatte, facile apertura, alta immunità Emc, isolamento, tensione d'uscita istantanea +/-4 V, alimentazione da +/-12 a 15 V formano le loro principali caratteristiche.
Costituiti da due parti apribili, allo scopo di essere montati attorno ad una barra o cavo senza interruzione delle connessioni esistenti, permettono misure di corrente senza contatto.
Semplici e veloci da installare, sono strumenti ideali per l'aggiornamento d'installazioni esistenti, quali alimentatori telecom.

readerservice.it 424 - 62

LETTORE DI TACCHE/SENSORE DI COLORE

Il nuovo lettore di tacche Sunx serie Lx100 di Matsushita permette di risolvere tutte le combinazioni possibili di tacca/sfondo con la scelta automatica del tipo di luce da utilizzare ed un tempo di risposta di soli 45 micros.
Consente inoltre di essere utilizzato anche come sensore di colore, funzione, questa, utile nel caso in cui lo sfondo sia variopinto o viceversa siano le tacche a variare.
È dotato di display a 4 cifre e tasti di memorizzazione/programmazione. La dimensione dello spot è di 1 x 5 mm e il range di 10 +/-3 mm.
Tutti i modelli hanno un grado di protezione Ip67 e si differenziano tra cablati o a connettore standard M12, con uscita Pnp o Npn.

readerservice.it 424 - 63

FPGA AD ALTA DENSITÀ RESISTENTI ALLE RADIAZIONI

Memec Unique ha annunciato la disponibilità dei campioni di Rtax250S, Rtax1000S e Rtax2000S, i tre dispositivi della nuova famiglia Actel di Fpga resistenti alle radiazioni ed ottimizzati per l'impiego in applicazioni spaziali.
Il comune denominatore di questi dispositivi programmabili è il sensibile incremento di prestazioni. Rtax2000S ha una densità di due milioni di gate di sistema equivalenti (o 250.000 gate Asic) e dispone di 288 K di Ram core, mentre per Rtax1000S la densità è pari a un milione di gate di sistema equivalenti (125.000 gate Asic) e la Ram disponibile è 162 K. Rtax250S mette invece a disposizione 250.000 gate di sistema equivalenti (30.000 gate Asic) e 54 K di Ram.

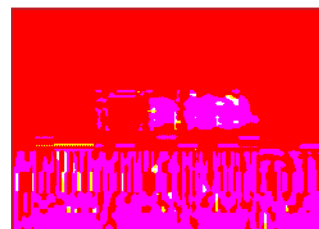
readerservice.it 424 - 64

POTENZIOMETRI DIGITALI

Xicor (Unique Memec) ha presentato due nuovi potenziometri digitali a bassissimo consumo che si rivolgono ai mercati che richiedono applicazioni in grandi volumi e con ingombri ridotti. Denominati X93155 e X93255, sono disponibili a prezzi altamente competitivi in formato singolo a doppio.
Dispongono di cursore a 32 posizioni con interfacce separate per aumentare o diminuire il valore di resistenza. Oltre a dissipare solo 2 microA alla massima corrente di standby, sono caratterizzati da una corrente nominale di funzionamento di 200 microA con una tensione di alimentazione di 5 V +/-10% e una non linearità differenziale inferiore a +/-0,2 Lsb.

readerservice.it 424 - 66

FOTORIFLETTORE

Njl5902R di New Japan Radio è un fotoriflettore

Cobp a montaggio superficiale, a fascio luminoso schermato, progettato principalmente per eseguire rilevazioni in svariate applicazioni. La superiore durata lo rende particolarmente adatto per usi in componenti audio per autoveicoli. Presenta dimensioni minime (2,7 x 1,9 x 0,8 mm) ed offre inoltre correnti scure ad un livello del 90 per cento inferiore a quello di Njl5901R esistenti. È in grado di resistere due volte a saldature a riflusso esenti da piombo a temperature fino a 260 gradi C ed è stato inoltre collaudato secondo cicli di 500 temperature (da -40 a +100 gradi C).

readerservice.it 424 - 67

SOLUZIONI MULTICANALE CON ATTENUATORE DI JITTER

Exar annuncia la prima famiglia di dispositivi che combinano funzionalità di LIU DS3/E3, Framer e attenuazione di jitter, sviluppata con l'avanzata tecnologia R3 di proprietà della società stessa. Questi dispositivi vengono introdotti sulla scia del grande successo del componente a canale singolo XRT79L71, rispetto al quale differiscono per il fatto di avere più canali, consentendo di ridurre l'ingombro sulla scheda e offrendo l'opportunità di aggiornare facilmente un prodotto alla versione a due, tre o quattro canali sulla medesima scheda. Le applicazioni di questa famiglia comprendono i sistemi di accesso, i sistemi digitali di cross-connect, i router, le stazioni base 3G, i DSLAM (digital subscriber line access multiplexer) e, più in generale, ogni apparecchiatura di commutazione di rete, comprese quelle sviluppate per i sistemi ATM, WAN e LAN.

readerservice.it 424 - 68

MICRO CON CORE SH-4A E CONTROLLORE PER BUS PCI

Renesas ha annunciato SH7780, un microprocessore che incorpora il core CPU SH-4A SuperH ed un controllore per bus PCI. Il dispositivo è caratterizzato da architettura a tre bus dedicati con prestazioni pari a 720 MIPS e 2.8 GFLOPS a 400 MHz, una combinazione che fornisce un miglioramento generale delle prestazioni del sistema. Ad esempio, SH7780 supporta sofisticati sistemi per riconoscimento e sintesi vocali, senza bisogno di DSP esterno. Il microprocessore è stato specificamente progettato per applicazioni multimediali ad elevate prestazioni, come sistemi di navigazione per automobili, giochi e prodotti elettronici digitali destinati all'utilizzo domestico.

readerservice.it 424 - 69

CONDENSATORI TRIMMER

La famiglia Sm571 di Temex concerne condensatori trimmer Gigahertz con variazione di capacitanza ultralineare, finalizzati ad applicazioni richiedenti sintonizzazioni molto fini per evitare aumenti improvvisi di capacitanza quando il rotore è vicino alla sua posizione massima o minima. I dispositivi in questione presentano un intervallo di capacitanza specificato da 0,8 a 6,0 pF su tensioni operative fino a 500 Vdc. La resistenza all'insolamento a tensioni operative normali è superiore a 10.000 Mohm con rotore situato sulla sua massima posizione. Sono stati progettati per sopportare urti di 100 g per 6 ms e vibrazioni di 60 g a 10-2000 Hz con un cambio di capacitanza inferiore all'1% rispetto al valore iniziale.

readerservice.it 424 - 70

DOPPI CONVERTITORI ANALOGICO-DIGITALI

Analog Devices ha reso disponibile un doppio convertitore analogico-digitale a 14 bit integrante due Adc a 14 bit su un chip singolo. Questo Ic innovativo è il dispositivo di bandiera di una nuova famiglia di doppi Adc ad alta velocità offerti da Analog Devices in una gamma di risoluzioni e velocità. Alloggiata in package da 9 x 9 mm, fornisce risoluzioni di 10, 12 e 14 bit a velocità da 20 a 105 Msps.
L'esclusiva combinazione di alta velocità, elevata risoluzione e packaging compatto consente di utilizzare soltanto un chip per conseguire le prestazioni di convertitori ad alta velocità multipli.

readerservice.it 424 - 73

COME OTTENERE MAGGIORI INFORMAZIONI

EONews offre il servizio "reader service" che vi consente, utilizzando l'apposito codice riportato alla fine di ogni news, di ricevere maggiori informazioni.

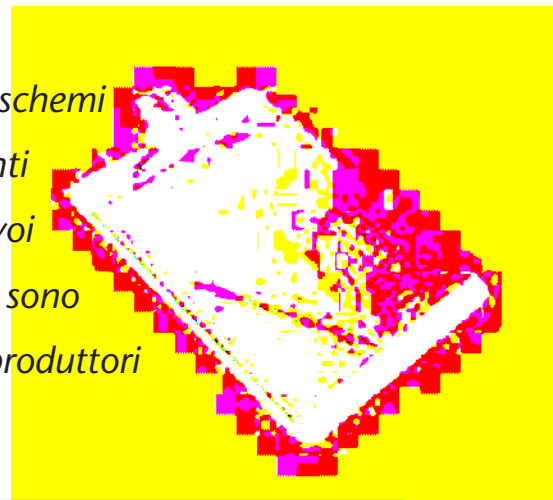
Visitate il sito **www.readerservice.it** e compilate la cartolina virtuale con i vostri dati, il numero della rivista, questo mese il **424**, ed i numeri di reader service presi dalle notizie che vi interessa approfondire.

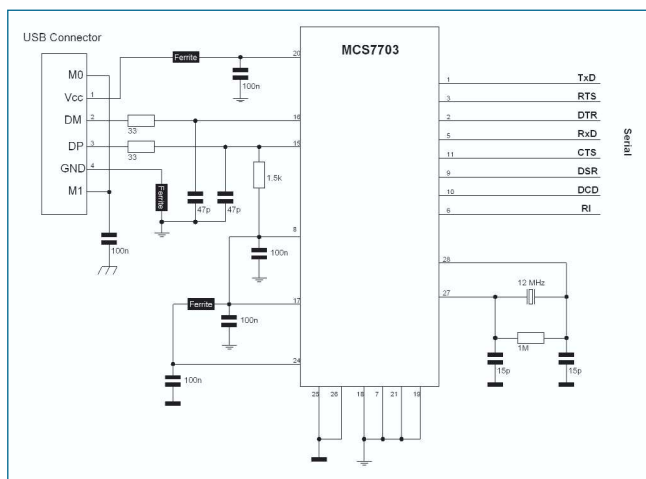
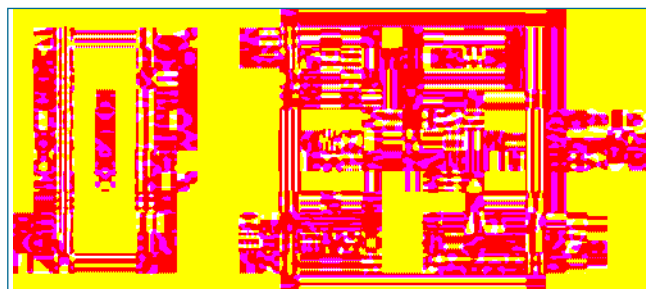
EONEWS provvederà, tempestivamente, a contattare le aziende interessate, che invieranno al vostro indirizzo tutta la documentazione disponibile.

Tutti i prezzi si intendono IVA inclusa

notepad

Dal blocco note di Fare Elettronica una raccolta di idee da tenere sempre a portata di mano

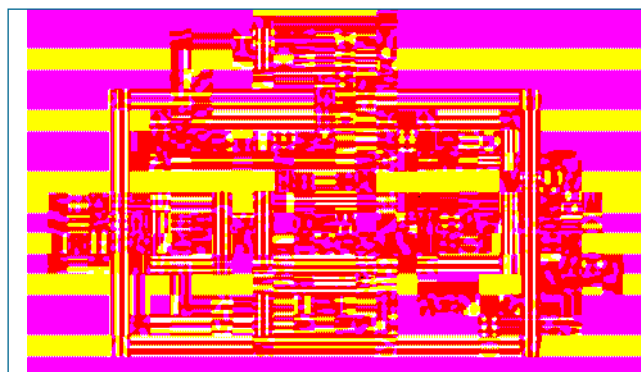
Questa rubrica ha lo scopo di fornire degli schemi applicativi o idee di progetto dei componenti elettronici più interessanti, selezionati per voi dalla redazione. Tutti gli schemi presentati sono elaborazioni di quelli ufficiali proposti dai produttori nella documentazione ufficiale.

CONVERSIONE USB/SERIALE CON MCS7703 — 29

Direttamente dalla Silicon Valley, la californiana Moschip Semiconductor fornisce l'integrato MCS7703 un chip in grado di operare la conversione Seriale/USB con l'ausilio di pochi componenti esterni. Dispone di un regolatore integrato a 3,3V pertanto può essere alimentato direttamente dal bus USB. I segnali disponibili sulla seriale sono Tx, Rx, RTS, CTS, DTR, DSR, DCD, RI. Consente la connessione con una EEPROM seriale esterna.
In figura il pin-out, lo schema a blocchi e lo schema applicativo.

PCF1171C: 4-DIGIT LCD CAR CLOCK — 30

Un orologio LCD da auto in un unico chip: PCF1171C di Philips. Questo circuito integrato è un orologio in grado di visualizzare ore e minuti direttamente su un display LCD passivo. Mediante switch esterni è possibile impostare ore e minuti e scegliere la modalità di visualizzazione 12h/24h.
In figura è riportato il pinout, l'identificazione dei segmenti del display ed uno schema applicativo.

PCF2112C: 4-DIGIT LCD DRIVER

31

Un driver per display LCD passivi a 4 cifre pilotabile mediante bus seriale a tre fili (CBUS). Gli ingressi sono compatibili CMOS/NMOS ed il dispositivo è realizzato in tecnologia CMOS Silicon Gate. La tensione di alimentazione può variare da 2.25V fino a 6V.
In figura il pin-out ed uno schema applicativo.

APPLICAZIONI EMBEDDED quarta parte

IMPLEMENTAZIONE DEL PROTOCOLLO MODBUS ASCII

Gianroberto Negri
g.negri@farelettronica.com

*Vediamo in questa puntata un'implementazione del protocollo **ModBus** in un suo sottoinsieme denominato ASCII, incapsulato nei messaggi UDP che saranno scambiati tra un'applicazione Master ed una Slave. Realizzeremo sia l'applicazione Master sia quella Slave affinché si possa osservare il reale scambio di messaggi/comandi tra loro. Implementeremo solo alcuni dei comandi visti nella precedente puntata, come in realtà avviene. Infatti, in un'apparecchiatura non vengono implementati tutti i comandi, ma solo quelli che servono per gestirla in modo ottimale.*

Abbiamo visto nella puntata precedente come è strutturato il protocollo **ModBus**, di come siano fatti i messaggi che transitano tra Master e Slave e come siano composti la maggior parte dei comandi, con i relativi esempi. Ora vedremo come implementarne alcuni, con riferimento al sottoinsieme ASCII.
Quanto realizzeremo sarà un ibrido poiché in realtà il sottoinsieme ASCII è normalmente utilizzato su linee seriali RS232/RS422/RS485 e non in rete sotto UDP o TCP/IP.
Bene, detto questo, immergiamoci decisamente nell'implementazione.
I comandi che saranno implementati sono:

- Codice Funzione/Comando 01 (0x01): Leggi Coils
- Codice Funzione/Comando 04 (0x04): Leggi Registro Ingresso (da 1 a 125 Registri)
- Codice Funzione/Comando 15 (0x0F): Scrivi Coils

Vediamo la relativa implementazione di ognuno e successivamente il tutto sarà messo insieme per realizzare l'applicazione Master e quella Slave.

Codice Funzione 01 (0x01): Leggi Coils
Questa funzione viene utilizzata per leggere lo stato dei Bits contigui che vanno da 1 a 2000 simultaneamente, con una unica richiesta. Nel PDU che è inviato allo Slave oltre ovviamente il Codice Funzione, viene anche specificato l'indirizzo da cui iniziare la lettura e per quanti Bits contigui.
Nella risposta lo Slave raggruppa i Bits in Bytes, divide cioè i Bits richiesti per 8 e li restituisce sotto forma di Bytes.
In figura 1 è riportato il suo algoritmo in forma grafica.

Codice Funzione 04 (0x04): Leggi Registro Ingresso
Questa funzione viene utilizzata per leggere da 1 a 125 Registri di Input da uno Slave. Il PDU specifica l'in-

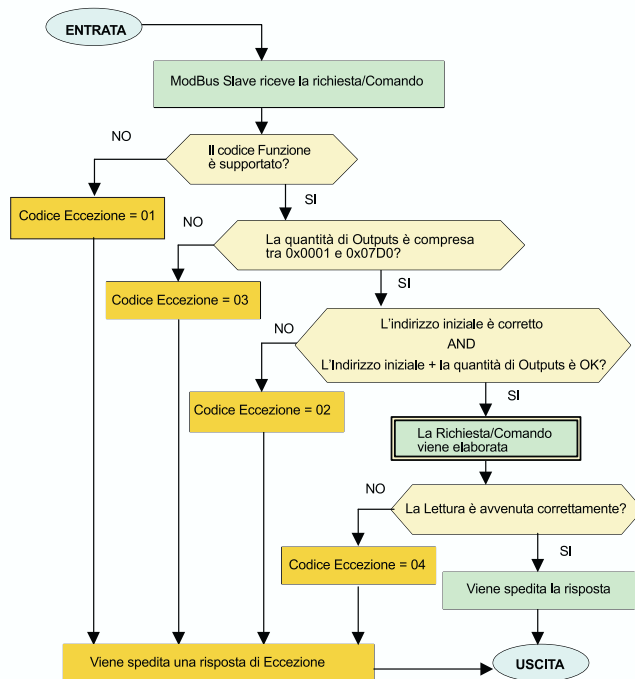



Figura 1: *Algoritmo Codice Funzione 01 Leggi Coils*

dirizzo del registro di partenza ed il numero di registri da leggere. L'indirizzo parte da 0. I dati contenuti nella risposta sono contenuti in un blocco di 2 Bytes per ogni registro, in cui nel primo è contenuta la parte alta HI e nel secondo la parte Bassa LOW.
In figura 2 è riportato il suo algoritmo in forma grafica.

Codice Funzione 15 (0x0F): Scrivi Coils multipli
Utilizzata per porre a ON o OFF i Coils di una sequenza di Coils. L'indirizzo iniziale parte da 0 come anche il Coil da modificare. Lo stato ON equivale a porre il corrispondente Bit a 1, mentre lo stato OFF a porre il Bit a 0.
La risposta dello Slave comprende il codice funzione, l'indirizzo di partenza e la quantità di Coils da modificare.
In figura 3 è riportato il suo algoritmo in forma grafica.

Sostanzialmente gli algoritmi si assomigliano, ed analizzandone il flusso possiamo osservare innanzitutto che riguardano lo Slave e la prima cosa che viene fatta è quella di attendere la ricezione di un comando inviato dal Master. Viene poi controllato che il codice funzione sia corretto e in caso contrario, viene generato un messaggio (risposta) di eccezione con codice di ritorno 01. Il passo successivo è quello di controllare che, la quantità sia compresa nell'intervallo previsto, se non lo è viene generata un'eccezione 03.
L'algoritmo prosegue e controlla che oltre la quantità sia corretto anche l'indirizzo iniziale da cui tale quan-

Figura 2: *Algoritmo Codice Funzione 04 Leggi Registri Ingresso*

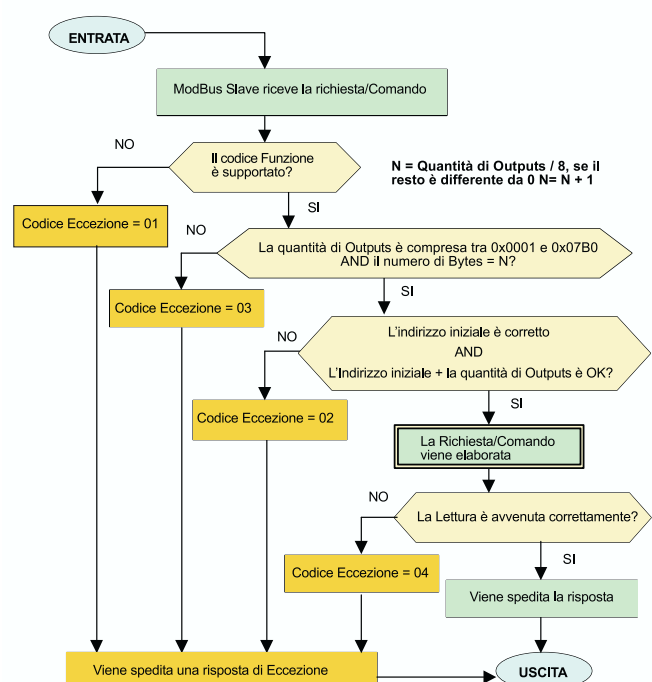



Figura 3: *Algoritmo Codice Funzione 15 Scrivi Coils multipli*

tità inizia appunto. Anche in questo caso se non lo è viene generato una eccezione 02. Come fase successiva, viene gestita l'elaborazione vera e propria e nell'ambito della stessa controllato che sia andata a buon fine, altrimenti viene generate una eccezione 04. La parte finale si preoccupa di rispedire al master i dati richiesti o un messaggio contenente l'eccezione verificatasi.

Esistono alcune differenze date dal tipo di richiesta e/o dati cui si fa riferimento, le vedremo ampiamente durante la spiegazione dei programmi Master e salve in cui sono implementati tali comandi.

Prima di andare avanti con la spiegazione di come sia stata fatta l'implementazione in Visual Basic 6.0, occorre scaricare i sorgenti dal sito di fare Elettronica, nel punto inerente a questo articolo, decomprimerli, aprire una sessione di VB 6.0 per ognuno di essi. Una sessione per il Master ed una per lo Slave.
In questa sede non verranno riportati tutti i sorgenti ma solo il diagramma a blocchi del master e dello Slave ed alcune parti significative. Questo perché il codice è molto ed occuperebbe troppe pagine se fosse riportato integralmente.

PROGRAMMA MASTER

Bene, detto questo vediamo ora il diagramma a blocchi del programma Master riportato in figura 4.

Nella programmazione che fa capo a Visual Basic e ad applicazioni che girino in sistemi a finestre, la gestione delle interazioni con l'utente sono del tipo Event Driven. Significa che, a parte l'inizializzazione, la gestione avviene per eventi che si "scatenano" alla pressione di un tasto od al Click del Mouse in una certa parte della finestra, in gergo *Form* presente a video. A meno che uno sviluppatore non decida di creare un loop che ciclicamente controlli ad esempio degli I/O all'interno della parte di inizializzazione. Quindi per inviare ad esempio un comando ModBus dal Master allo Slave, come nel nostro caso, occorre che l'utente clicchi ad esempio su di un tasto.
Sostanzialmente il programma si suddivide in tre blocchi funzionali:

- Inizializzazione
- Comandi
- Risposte

L'inizializzazione (Form_load()): Viene richiamata quando il programma viene lanciato e viene creato il Form MasterA. Si occupa di impostare i valori previsti (scelte) nelle varie ComboBox presenti nella videata (Form) Master. Tali valori permetteranno all'utente di sceglierli senza preoccuparsi di doverli immettere.
Successivamente è impostato il protocollo UDP per colloquiare con il programma Slave che, girerà contemporaneamente al Master.

Comandi (InviaOK_Click()): Viene richiamata al Click del tasto **Invia Comando**. Si occupa di gestire le scelte effettuate dall'utente, in merito al comando da inviare ed ai suoi parametri impostati mediante le ComboBox e di inviare il comando. Esistono due possibilità riguardanti l'invio dei comandi allo Slave. *Comandi preparametrizzati e comandi che utilizzano i parametri impostabili mediante le ComboBox*. Il passaggio tra una possibilità e l'altra avviene mediante la CheckBox **Usa Parametri**. Spuntandola la costruzione del comando avviene mediante appunto i parametri disponibili a video, altrimenti vengono reimpostati a programma. Questa possibilità é stata data per fare un po' di test riguardo il colloquio tra Master e Slave e vedere come quest'ultimo reagisce.

Risposte (udpPeerA_DataArrival): Viene richiamata all'arrivo di una risposta dallo Slave. Si occupa di gestire le risposte ed è strutturata nel seguente modo:
- Sono dimensionate le variabili necessarie.
- È acquisita la risposta dello Slave interessato ed è salvata nella lista Risposte del video.
- Viene controllato che siano arrivati almeno 7 caratteri altrimenti viene scartata la risposta.
- Vengono estratte alcune parti che compongono la risposta tra cui il controllo LRC che viene controllato. Se a questo punto è tutto ok, in base al codice di ritorno del comando, è richiamata la funzione prevista.

ANALISI FUNZIONI MASTER

Dal blocco Comandi sono richiamate le seguenti funzioni:

- **ReadCoils:** implementazione del comando **01 (0x01) Read Coils.**

DIAGRAMMA A BLOCCHI MASTER

Event Driven

Inizializzazione
Form_load()

Comandi
InviaOK_Click()

Risposte
udpPeerA_DataArrival()

Impostazione valori di scelta nelle CoboBox
Attivazione protocollo UDP con 127.0.0.1

Scelta comando da inviare in funzione delle impostazioni delle ComboBox.
Comandi previsti:
ReadCoils
ReadInputRegisters
WriteMultipleCoils

Dimensionamento variabili
Acquisizione risposta dallo Slave
Test che il numero di caratteri sia => di 7
Estrazione partie funzionali dalla risposta
Controllo LRC
Controllo Slave
Gestione risposte a Comandi previsti:
RispReadCoils
RispReadInputRegisters
RispWriteMultipleCoils

HexToDec
Conversione da Esadecimale a Decimale

Calcolo LRC

Dimensionamento variabili
Caricamento valori (Parametri) dalle ComboBox
Creazione Comando
Calcolo LRC
Invio Comando
Salvataggio Comando in lista video.

Calcolo LRC

HexToDec
Conversione da Esadecimale a Decimale

Dimensionamento variabili
Caricamento valori (Parametri) dalle ComboBox
Creazione Comando
Calcolo LRC
Invio Comando
Salvataggio Comando in lista video.

Calcolo LRC

HexToDec
Conversione da Esadecimale a Decimale

Dimensionamento variabili
Caricamento valori (Parametri) dalle ComboBox
Creazione Comando
Calcolo LRC
Invio Comando
Salvataggio Comando in lista video.

CostruisciWriteMultipleCoils
Costruzione dati per scrittura

BinToDEc
Conversione da Decimale a Binario

Calcolo LRC

HexToDec
Conversione da Esadecimale a Decimale

Dimensionamento variabili
Estrazione Numero di Bytes ricevuti
Estrazione Risposta
Attivazione Visualizzazione con i valori ricevuti dallo Slave
Salvataggio Risposta in lista video.

HexToDec
Conversione da Esadecimale a Decimale

FormatBinary2
Conversione da Decimale a Binario

Dimensionamento variabili
Estrazione Numero di Bytes ricevuti
Estrazione Risposta
Attivazione Visualizzazione con i valori ricevuti dallo Slave
Salvataggio Risposta in lista video.

HexToDec
Conversione da Esadecimale a Decimale

Dimensionamento variabili
Estrazione Numero di Bytes ricevuti
Estrazione Risposta
Salvataggio Risposta in lista video.

Figura 4: *Diagramma a blocchi programma Master*

- **ReadInputRegisters:** implementazione del comando **04 (0x04) Read Input Registers.**
- **WriteMultipleCoils:** implementazione del comando **15 (0x0F) Write Multiple Coils.**

Analizziamole, iniziando dalla **ReadCoils**.
In essa per prima cosa vengono dimensionate e/o definite le variabile occorrenti alla funzione, poi in base al valore della variabile CmdOK, vengono utilizzate o quelle provenienti dalle ComboBox oppure quelle che vengono caricate con valori predefiniti. In quest'ultimo caso i valori sono:

- **3** per lo Slave
- **0** per l'indirizzo iniziale
- **8** per il numero di Coils da leggere dall'indirizzo impostato

Fatto questo, viene calcolato il controllo LRC, viene costruito il comando e viene salvato nella Lista dei comandi presente a video.

La funzione **ReadInputRegisters** è strutturata nella stessa maniera, vengono per prima cosa definite le variabili, precaricate quelle provenienti dalle ComboBox e, in base al valore di CmdOK, utilizzate quelle delle ComboBox o quelle precaricate i cui valori sono:

- **3** per lo Slave
- **8** per l'indirizzo iniziale
- **1** per il numero di registri da leggere dall'indirizzo impostato

È poi calcolato il controllo LRC, costruito il comando, inviato e salvato nella List dei comandi presente a video.

La funzione **WriteMultipleCoils** è anch'essa similmente strutturata, cambiano soltanto i valori precaricati che sono:

- **3** per lo Slave
- **0** per l'indirizzo iniziale
- **8** per il numero di Coils da leggere dall'indirizzo impostato

È questa istruzione che si occupa di convertire il valore binario dei Coils impostati a video in un valore, prima decimale e poi esadecimale:

```
Coils = Right("0000" + Hex(BinToDec(Coils)), 4)
```

Il valore contenuto nella variabile Coils proviene da un'altra funzione la **CostruisciWriteMultipleCoils** che si occupa di leggere i valori impostati a video nelle CheckBox e di convertirli in una stringa Binaria contenente ovviamente i valori 1 o 0 a secondo che le CheckBox abbiano o meno il segno di spunta.
Poiché solo valori esadecimali sono accettati, la stringa Coils è convertita mediante la funzione intermedia **BinToDec** e la funzione di sistema Hex in tali valori.
La parte finale calcola il valore del controllo LRC, costruisce il comando, lo spedisce e lo salva nella lista dei comandi presente a video.

Dal blocco Risposte sono richiamate le seguenti funzioni:

- **RispReadCoils**: implementazione della risposta inerente al comando **Read Coils**.
- **RispReadInputRegisters**: implementazione della risposta inerente al comando **Read Input Registers**.
- **RispWriteMultipleCoils**: implementazione della risposta inerente al comando **Write Multiple Coils**.

La prima funzione è **RispReadCoils**. In essa vengono come prima cosa definite le variabili necessarie, poi viene estratto dalla risposta il numero di Bytes di cui è composta la parte dati della risposta ed infine i dati della risposta veri e propria. I dati così prelevati sono utilizzati per cambiare colore ai Led presenti nel Form MasterA che identificano gli stati presenti al momento della lettura sullo Slave.

Segue la funzione **RispReadInputRegisters**. In essa, come nella precedente, vengono definite come prima cosa le variabili, poi estratto il numero di Bytes della parte dati del messaggio ed infine i dati veri e propri. Anche in questo caso quanto acquisito è visualizzato a video. Poiché si tratta di dati analogici viene visualizzato in un campo apposito il valore corrispondente.

Infine la funzione **RispWriteMultipleCoils**, si limita unicamente a controllare che la risposta sia conforme.

FUNZIONI ACCESSORIE AL MASTER ED ALLO SLAVE

Prima di passare alla descrizione del programma Slave, ci soffermeremo ora su alcune funzioni che sono state utilizzate sia dal Master sia dallo Slave per gestire conversioni da un formato all'altro o per il calcolo del controllo LRC. Di seguito l'elenco delle stesse:

- **HexToDec**
  converte da esadecimale 16 Bit a Decimale
- **HexToDec2**
  converte da esadecimale 8 Bit a Decimale
- **FormatBinary**
  converte da Decimale a Binario 16 Bit
- **FormatBinary2**
  converte da Decimale a Binario 8 Bit
- **BinToDec**
  converte da Binario 16 Bit a Decimale
- **BinToDec2**
  converte da Binario 8 Bit a Decimale
- **LRC**
  effettua il calcolo LRC

Tutte queste sono funzioni liberamente utilizzabili nelle Vostre applicazione e costituiscono una sorta di libreria pronta all'uso e testata in quanto al funzionamento.
Vediamo ora per ognuna quale formato dati passare ed in quale formato, una volta convertiti gli stessi, sono restituiti.

HexToDec accetta in ingresso un valore in **esadecimale** (nel formato 000F, ABCE, 0012, 0001) caricato in una variabile di tipo stringa o posto tra doppi apici (come "0F01"). Restituisce un valore di tipo **Long** che può andare da 0 a 65535.

HexToDec2 accetta in ingresso un valore in **esadecimale** (nel formato 0F, CE, 12, 01) caricato in una variabile di tipo stringa o posto tra doppi apici (come "0F"). Restituisce un valore di tipo **Long** che può andare da 0 a 255.

FormatBinary accetta in ingresso un valore di tipo **Long** (da 0 a 65535) e restituisce una stringa di 16 caratteri (come 0000111100001010).

FormatBinary2 accetta in ingresso un valore di tipo **Long** (da 0 a 255) e restituisce una stringa di 8 caratteri (come 00001010).

BinToDec accetta in ingresso una stringa da 16 caratteri massimo (come 0000111100001010) e restituisce un valore di tipo **Long** che può andare da 0 e 65535.

BinToDec2 accetta in ingresso una stringa da 16 caratteri massimo (come 00001010) e restituisce un valore di tipo **Long** che può andare da 0 a 255.

LRC accetta in ingresso una stringa di 255 caratteri in formato esadecimale da 2 caratteri per cifra, come ad esempio FF0F01. Da tale stringa, le cifre verranno prelevate a doppiette (due cifre per volta), quindi nel caso dell'esempio FF0F01, le cifre saranno nell'ordine FF 0F 01. Tali cifre verranno sommate tra di loro, poi al totale verrà sommato il valore esadecimale FF ed al tutto il valore esadecimale 1. Restituisce un valore esadecimale di 4 caratteri che corrisponde al valore LRC richiesto.

PROGRAMMA SLAVE

Vediamo ora il diagramma a blocchi del programma Slave riportato in figura 5.

Contrariamente al programma Master, lo Slave non ha capacità decisionali autonome, si limita ad eseguire i comandi che gli provengono dal Master, elaborarli e restituire i risultati richiesti.
Si suddivide in due blocchi:

- Inizializzazione
- Gestione Comandi provenienti dal Master

L'inizializzazione Viene richiamata quando il programma viene lanciato e viene creato il Form SlaveB. Si occupa di impostare il protocollo UDP per colloquiare con il programma Master che girerà contemporaneamente allo Slave.

Gestione Comandi provenienti dal Master (udpPeerB_DataArrival)
Questa funzione viene richiamata all'arrivo di un comando dal Master. Si occupa di gestire i comandi ricevuti ed è strutturata nel seguente modo:

- Sono dimensionate le variabili necessarie.
- Viene acquisita la richiesta e viene salvata nella lista Comandi del video.
- Viene controllato che siano arrivati almeno 7 caratteri altrimenti viene scartato.
- Vengono estratte le parti funzionali che lo compongono, tra cui il controllo LRC che viene verificato. Se a questo punto è tutto ok, in base al codice del comando, è richiamato il blocco di funzioni previsto.

Dal blocco Gestione Comandi provenienti dal Master sono richiamate le seguenti funzioni:

- Implementazione del comando **01 (0x01) Read Coils**.
  - ERispReadCoils
  - RispReadCoils
  - RispReadCoilsECC
- Implementazione del comando **04 (0x04) Read Input Registers**.
  - ERispReadInputRegisters
  - RispReadInputRegisters
  - RispReadInputRegistersECC
- Implementazione del comando **15 (0x0F) Write Multiple Coils**.
  - ERispWriteMultipleCoils
  - RispWriteMultipleCoils
  - RispWriteMultipleCoilsECC

Prima di analizzarle occorre fare una premessa, mentre nel Master l'invio del comando e la ricezione della risposta sono gestiti da due distinti eventi, nello Slave a causa del fatto che lo stesso non può arbitrariamente trasmettere, il tutto è gestito da un unico evento che si occupa di:

- Controllare il comando pervenuto.
- Gestire il comando ricevuto elaborandolo e generando la risposta per il Master.
- Gestire l'eventuale risposta di errore o Eccezione in caso di comando non corretto o con parametri non validi.

Sostanzialmente la gestione di ogni comando è la stessa, come si può notare osservando il diagramma a blocchi dello Slave in figura 5. Riguardo ai parametri ogni Slave ha i suoi. Questo vuol affermare che gli Slave possono essere differenti tra loro. Anche in questa simulazione il programma Slave ha i suoi parametri e, se il Master fa delle richieste che non rientrano in quelli previsti, viene generata un'eccezione con il codice di ritorno inerente al parametro sbagliato. Se ad esempio viene fatta una richiesta da parte del Master in cui il comando è sintatticamente corretto ma i suoi parametri (Indirizzo iniziale, numero di Coils o Registri) non sono uguali a quelli previsti, lo Slave risponde con un'eccezione il cui codice indica quale è il parametro errato. Il fatto di aver creato una diversificazione nella spedizione dei comandi da parte del

Figura 5: *Diagramma a blocchi programma Slave*

Master ha lo scopo di inviare un comando impostato a programma con i parametri conformi a quelli dello Slave, oppure, di testare come lo Slave risponda a parametri non corretti.

IMPLEMENTAZIONE DEL COMANDO 01 (0X01): READ COILS

Vediamo ora i blocchi di funzioni utilizzate, iniziando da quelle riferite al comando **ReadCoils**, quindi con **l'ERispReadCoils**. Questa ha il compito di controllare se sono state fatte delle richieste con parametri non conformi. Il primo è un controllo formale e non riguarda direttamente come è configurato lo Slave, viene controllato che il numero di Coils richiesti sia compreso tra 1 e 2000 ed in caso contrario viene generata una eccezione 03. Viene poi controllato che la somma dell'indirizzo iniziale da cui partire ed il numero di Bytes richiesti siano conformi con la configurazione dello Salve. Questo accetta solo come indirizzo iniziale 0 e come numero massimo di Bytes 1. Anche in questo caso per valori non corretti viene generata un eccezione 02. L'ultimo controllo simula il fatto che, anche se tutti i parametri sono corretti, ma per qualche motivo la lettura non è possibile, viene generata un eccezione 04. Per poter attivare la simulazione occorre spuntare la CheckBox *Blocca Lettura/Scrittura* presente nel Form dello Slave. Se tutti i controlli hanno dato esito negativo, viene restituito il valore 0 che da l'abilitazione a procedere con la gestione del comando ricevuto mediante la funzione **RispReadCoils**, oppure, con la funzione **RispReadCoilsECC** risponde al master con l'eccezione prevista.

Proseguiamo con la funzione **RispReadCoils** che si occupa di gestire le richieste/comandi provenienti dal Master, elaborandole e restituendo una risposta con i dati richiesti. In essa vengono come prima cosa definite le variabili necessarie, poi viene costruita la risposta per il Master. Per la parte dati costituita dalla variabile Coils, il suo contenuto proviene da un'altra funzione la **CostruisciRispostaReadCoils** che si occupa di leggere i valori impostati a video nelle CheckBox e di convertirli in una stringa Binaria da 8 Bit, contenente ovviamente i valori 1 o 0 a secondo che le CheckBox abbiano o meno il segno di spunta. Poiché sono accettati solo valori esadecimali, la stringa Coils è convertita mediante la funzione intermedia **BinToDec2** e la funzione di sistema Hex in tali valori:

```
NCoilsStr = Right("00" + Hex(BinToDec2(Coils)), 2)
```

La parte finale calcola il valore del controllo LRC, costruisce la risposta, la spedisce e la salva nella lista delle risposte presente a video.
La funzione **RispReadCoilsECC** ha il compito di spedire le eccezioni quando accadono. In essa vengono definite le variabili, costruita la risposta per il master, calcolato il controllo LRC, spedito il tutto salvandolo anche nella Lista del video (Form) inerente alle risposte. È interessante osservare l'istruzione:

```
Eccezione = Right("00" + Hex(Test + 128), 2)
```

In essa al valore proveniente dalla funzione **ERispReadCoils**, contenuto nella variabile Test, viene sommato il valore 128 (80 in esadecimale), come da specifiche, per individuare le eccezioni che provengono dal comando **Read Coils**.

IMPLEMENTAZIONE DEL COMANDO (0X04): READ INPUT REGISTERS

Passando al prossimo blocco di funzioni utilizzate, troviamo quelle riferite al comando **Read Input Registers**.

La prima **ERispReadInputRegisters** come di consueto è quella adibita la controllo funzionale e parametrico. Viene controllato che il numero di Registri richiesti sia compreso tra 1 e 120 ed in caso contrario viene generata una eccezione 03. Viene poi controllato che la somma dell'indirizzo iniziale da cui partire ed il numero di Registri richiesti siano 8 per indirizzo iniziale e 1 come massimi registri disponibili. Anche in questo caso per valori non corretti viene generata un eccezione 02. L'ultimo controllo è identico all'altro simulando il fatto che, anche se tutti i parametri sono corretti ma per qualche motivo la lettura non è possibile, genera una eccezione 04.
Proseguiamo con la funzione **RispReadInputRegisters** che si occupa di gestire ed elaborare le richieste, restituendo una risposta con i dati richiesti. In essa vengono come prima cosa definite le variabili necessarie, poi viene costruita la risposta per il Master. Come per la precedente la funzione:

```
Public Function
CostruisciRispReadInputRegisters() As String
   CostruisciRispReadInputRegisters = Right("0000" & Hex(Val(SlaveB.Word0.Text)), 4)
End Function
```

Si occupa di leggere il valore impostato a video dall'utente, di convertirlo in un valore esadecimale a 16 Bits e di passarlo alla funzione **RispReadInputRegisters** , la quale provvede a calcolare il valore del controllo LRC, costruire la risposta, spedire e salvare nella lista delle risposte presente a video.

La funzione **RispReadInputRegistersECC** spedisce al Master l'eccezione che si è verificata.
L'istruzione:

```
Eccezione = Right("00" + Hex(Test + 132), 2)
```

utilizzando il valore proveniente dalla funzione **ERispReadInputRegisters**, contenuto nella variabile Test, somma il valore 132 (82 in esadecimale), come da specifiche, per individuare le eccezioni che provengono dal comando **Read Input Registers**.

IMPLEMENTAZIONE DEL COMANDO 15 (0X0F): WRITE MULTIPLE COILS

Passando al blocco successivo di funzioni utilizzate, troviamo quelle riferite al comando **Write Multiple Coils.**

La prima **ERispWriteMultipleCoils** controlla che, il numero di Coils richiesti sia compreso tra 1 e 1968 ed, in caso contrario, genera una eccezione 03. Viene poi controllato che la somma dell'indirizzo iniziale da cui partire ed il numero di Registri richiesti siano: 0 per l'indirizzo iniziale e 8 come massimi Coils disponibili. Anche in questo caso per valori non corretti viene generata un eccezione 02. L'ultimo controllo è identico agli altri due blocchi funzioni, simulando il fatto che, anche se tutti i parametri sono corretti, se per qualche motivo la lettura non è possibile genera una eccezione 04.
Proseguiamo con la funzione **RispWrite-MultipleCoils** che si occupa di gestire la richiesta elaborandola. Questa estrae il numero di Bytes, di cui è composta la parte dati della richiesta, ed i dati della richiesta veri e propri. I dati così prelevati vengono utilizzati per cambiare colore ai Led presenti nel Form SlaveB e che identificano gli stati presenti al momento della richiesta sul Master.

La funzione **RispWriteMultipleCoilsECC** spedisce al Master l'eccezione che si è verificata.
L'istruzione:

```
Eccezione = Right("00" + Hex(Test + 143), 2)
```

utilizzando il valore proveniente dalla funzione **ERispWriteMultipleCoils**, contenuto nella variabile Test, somma il valore 143 (8F in esadecimale), come da specifiche, per individuare le eccezioni che provengono dal comando **Write Multiple Coils**.
In figura 6 possiamo osservare il programma Master all'opera ed in figura 7 lo Slave.

CONCLUSIONI

Bene, anche per questa puntata abbiamo terminato, nella prossima continueremo a parlare del MODBUS, implementando la parte emulatore PLC in Visual Basic.

Figura 6: Il programma Master all'opera

Figura 7: Il programma Slave all'opera

GLI ALIMENTATORI SWITCHING

prima parte

INTRODUZIONE

di Romano Bernarducci
r.bernarducci@farelettronica.com

Il corso partirà dalla progettazione e realizzazione dei vari blocchi "discreti" che compongono un alimentatore switching, evitando inizialmente di utilizzare gli schemi di applicazione di circuiti integrati commerciali.

Non inizierò con schemi di applicazione di circuiti integrati disponibili su qualunque datasheet, ma realizzando i vari blocchi funzionali con componenti separati. Questo approccio particolarmente didattico, vi permetterà non solo di progettare autonomamente, ma anche di comprendere le cause dei più comuni malfunzionamenti degli switching. Solo più avanti presenterò circuiti applicativi di IC commerciali, sempre però facendo prevalere il lato didattico su quello puramente teorico/pratico.
Nel corso parlerò di:

- Differenze tra alimentatori switching e lineari.
- Tipologie standard: buck, boost, buck/boost, inverting.
- Circuiti di controllo: PFM, PWM Voltage Mode, PWM Current Mode.
- Utilizzo di IC standard.
- Tipologia flyback, trasformatori.
- Alimentatori multi-uscita.
- Alimentatori isolati da rete.
- Analisi di un alimentatore da rete per PC.

Per quasi tutti gli argomenti presenterò opportune formule per il progetto autonomo di alimentatori similari.

INTRODUZIONE

Qualsiasi circuito elettronico ha necessità, per il proprio corretto funzionamento, di una o più tensioni continue, in genere stabili o persino variabili, sempre però entro limiti ben definiti.
Queste tensioni si ottengono a partire da una sorgente di energia di vario tipo (rete elettrica, batterie, celle solari, ecc.), per mezzo di opportuni circuiti trasformatori/convertitori, che definiamo genericamente alimentatori.

ALIMENTATORI LINEARI

Non mi dilungherò più di tanto su questo argomento: considerate la figura 1 che riporta lo schema di principio di un alimentatore lineare con regolazione serie (series o pass regulation). Il circuito di controllo varia la resistenza in serie al carico finché la tensione di uscita non coincide con quella desiderata e la mantiene costante al variare del carico, della tensione di ingresso e dei parametri del circuito.

La resistenza variabile è in realtà costituita da dispositivi a semiconduttore come i transistor BJT e/o i MOSFET.
Per le informazioni di progetto relative a questo tipo di alimentatori rimando alla serie di articoli di *Nico Grilloni* sugli stabilizzatori di tensione (Fare Elettronica 227 e seguenti).
Le limitazioni degli alimentatori lineari sono principalmente due: la necessità di una tensione di ingresso sempre superiore a quella di uscita (o equivalentemente, l'impossibilità di generare una tensione superiore a

Figura 1: *Schema di principio di alimentatore lineare con regolazione serie.*

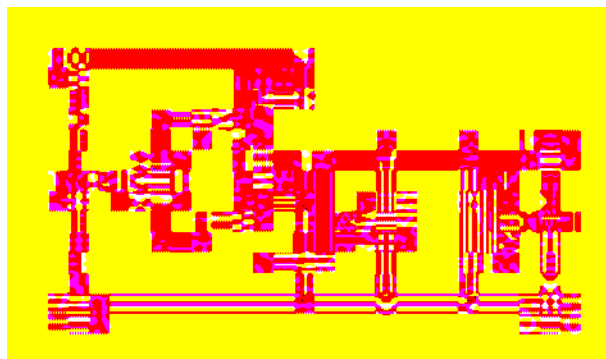



Figura 2: *Schema di principio di alimentatore switching di tipo charge-pump (invertente).*

quella di ingresso) e la ridotta efficienza.
Ricordo che l'efficienza di un qualsiasi alimentatore, lineare o switching, è indicata con la lettera greca η (eta) ed è definita nella formula 1:

Formula 1
$\eta = P_{OUT} / P_{IN} * 100$
(P_{OUT} e P_{IN} in Watt, η in percentuale)

dove P_{OUT} è la potenza erogata al carico e P_{IN} è la potenza assorbita dall'ingresso. La differenza $P_{IN} - P_{OUT}$, sempre maggiore di zero, viene dissipata in calore dall'elemento serie di figura 1. Poiché la potenza elettrica P è pari al prodotto della tensione V per la corrente I, ne deriva che sono necessarie alette di raffreddamento sempre più grandi all'aumentare della corrente richiesta dal carico e del "salto di tensione" tra ingresso e uscita ($V_{IN} - V_{OUT}$). A favore degli alimentatori lineari va riconosciuta comunque la facilità di progetto e di realizzazione, l'assenza di rumore legato alla commutazione, il ridotto numero di componenti, l'elevata affidabilità.
Quando è necessario o consigliabile passare agli switching?
Sempre, se la tensione di uscita è superiore a quella di ingresso oppure se è negativa rispetto all'ingresso. Nel caso "classico" in cui la tensione di uscita è inferiore a quella di ingresso, consiglio di utilizzare l'alimentatore lineare finché questo non necessita di aletta di raffreddamento o al massimo, di una "piccola" aletta. Per fare un esempio, supponiamo di avere a disposizione una tensione di 5V per alimentare un microprocessore che assorbe 400mA massimi a 3,3V. La dissipazione nell'elemento serie di un regolatore lineare è pari a (5 – 3,3) * 0,4 = 0,68W, facilmente dissipabili da qualsiasi regolatore integrato in contenitore TO220, senza aletta.

ALIMENTATORI SWITCHING

Per dare un'idea grossolana ma efficace, gli alimentatori switching funzionano tutti prendendo energia dall'ingresso, immagazzinandola in un apposito "serbatoio" non dissipativo, per poi rilasciarla al carico a tensione differente. Questo procedimento ciclico a due fasi carica-scarica, avviene molte volte al secondo, da cui il nome di alimentatori a commutazione. La trasformazione di tensione avviene sfruttando opportune modalità che costituiscono l'argomento principale del corso. Se i componenti elettronici di contorno e il "serbatoio" fossero ideali, cioè non dissipativi, avremmo un'efficienza η pari al 100%. Questo è fisicamente impossibile, tuttavia switching particolari arrivano ad efficienze superiori al 95%. Il "serbatoio" di energia non dissipativo, può essere indifferentemente costituito da condensatori e da induttanze, o da entrambi. La disponibilità di due diversi componenti porta ad una differenziazione degli switching in due categorie, i cosiddetti "charge-pump" (pompa di carica), che utilizzano come "serbatoio" i condensatori, e i più versatili magnetici che utilizzano induttanze e trasformatori.

ALIMENTATORI SWITCHING CHARGE-PUMP

Lo schema di principio di uno switching charge-pump è riportato in figura 2.
Questo circuito genera una tensione negativa di valore opposto alla tensione di ingresso.

Durante la fase 1, il condensatore C1 si carica a V_{IN}. Nella fase 2, si commutano contemporaneamente gli interruttori in figura, quindi il condensatore C1 viene "scaricato" sul condensatore di uscita C2, con la polarità invertita (cioè il pin di C1 carico positivamente viene collegato a GND, e l'altro pin, negativo rispetto a massa, si scarica su C2). In realtà la scarica non è completa, ma c'è una ripartizione di carica che dipende dalla differenza di tensione e di capacità tra C1 e C2.
La frequenza di commutazione è in genere limitata a pochi kHz. Su questo principio si basa il ben noto circuito integrato 7660, prodotto da varie case costruttrici. L'utilizzo dei charge-pump è limitato a carichi di alcune decine di mA, e comunque la tensione d'uscita non è ben stabilizzata ma varia all'aumentare del carico (la resistenza d'uscita è notevole, dell'ordine delle decine di ohm).
Quando utilizzare i charge-pump?
Ad esempio, per alimentare in duale un op-amp avendo a disposizione solo una tensione positiva.
Gli alimentatori charge-pump sono sicuramente più rumorosi dei lineari, ma molto meno di uno switching magnetico, quindi sono preferibili per l'utilizzo in circuiti sensibili di bassa potenza.

ALIMENTATORI SWITCHING "MAGNETICI"

Gli switching di tipo magnetico utilizzano come "serbatoio" di energia un'induttanza o un trasformatore. Per ora trascuriamo il trasformatore, che sarà necessario solo per i circuiti avanzati. Dal punto di vista descrittivo, possiamo considerare l'induttanza come il componente "duale" del condensatore. Intendo con "duale" un componente che si comporta elettricamente come l'altro, ma con **la tensione e la corrente scambiati di ruolo**.
Mi spiego meglio: è noto che un condensatore inizialmente scarico (V = 0), attraverso il quale facciamo scorrere una determinata corrente, non si carica "istantaneamente", ma aumenta gradualmente la sua tensione nel tempo.
Per la "duale" induttanza, invertendo i ruoli di I e V, si ottiene che, partendo inizialmente con corrente nulla (I = 0), e applicando ai capi una tensione, la corrente non varierà "istantaneamente" ma aumenterà gradualmente nel tempo. La formula 2 sintetizza quanto detto:

Figura 3: *Corrente nell'induttanza nel tempo.*

Formula 2
$\Delta I_L = (V_L / L) * \Delta t$
(ΔI_L in Ampere, Δt in secondi, L in Henry, V_L in Volt)

Come noto, il simbolo Δ (delta) indica una variazione o un intervallo, quindi ΔI_L indica la variazione di corrente nell'induttore, nell'intervallo Δt. La formula 2 vale solo per induttanze ideali, ma si può utilizzare, con le opportune cautele, anche nei casi reali. Si noti che la corrente aumenta linearmente nel tempo e sale tanto più rapidamente quanto più è alta la tensione di ingresso e quanto più è bassa l'induttanza. In figura 3 sono visibili i grafici della corrente per varie induttanze e tensioni applicate.

Quanta energia possiamo immagazzinare nel nostro "serbatoio" magnetico? Vedremo che anche questo parametro è molto importante nei futuri calcoli di progetto; senza troppe spiegazioni, prendete per buona la formula 3:

Formula 3
$E_L = 0,5 * L * I_L^2$
(E_L in Joule, L in Henry, I_L in Ampere)

Ricordate che 1 Joule = 1 Watt / sec., cioè un induttore "carico" di 1J di energia magnetica, potrebbe alimentare un carico di 1 Watt per un tempo di 1 secondo. Per fare un esempio grossolano ma che può darvi un'idea, consideriamo l'induttanza che useremo per il nostro primo switching, una 220µH - 0,6 Ampere max. (catalogo RS 233-5235). "Caricandola" alla massima corrente ammessa, l'energia immagazzinata sarà pari a: $E_L = 0,5 * 220 * 10^{-6} * 0,6^2 = 39,6µJ$. Questa energia riuscirebbe ad alimentare il nostro carico di 1W, per la bellezza (!) di circa 40µs. Ma poiché la prerogativa degli alimentatori switching è quella di ricaricare questo piccolo "serbatoio" magnetico migliaia di volte al secondo, l'energia a disposizione del carico aumenterà notevolmente.

TIPOLOGIE DI ALIMENTATORI SWITCHING "MAGNETICI"

Si intende per tipologia, una particolare connessione degli elementi magnetici e dei componenti elettronici dell'alimentatore switching, che permette la trasformazione di tensione (riduzione, incremento o inversione).
Ciascuna tipologia porta con se alcuni vantaggi e svantaggi, ed è caratterizzata da differenti formule di progetto.

TIPOLOGIA BOOST

La tipologia boost ("incrementatore") fornisce, come intuibile, una tensione di uscita sempre superiore a quella di ingresso.
Questo è uno dei vantaggi degli alimentatori switching messo in evidenza precedentemente. Lo schema di principio di un convertitore boost consiste di due interruttori, un'induttanza e un condensatore disposti come in figura 4.
Trascuriamo per questa puntata

Figura 4: *Schema di principio di un alimentatore switching di tipo boost.*

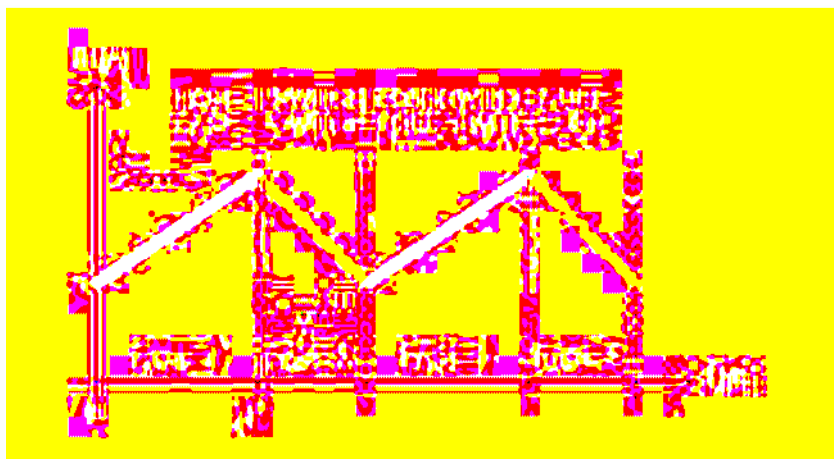



Figura 5: *Corrente nell'induttanza (tipologia boost)*

il sistema di controllo, e supponiamo che tutti i componenti siano ideali, cioè che l'induttanza abbia una resistenza in continua R_L pari a zero, il condensatore non abbia "perdite" (parleremo più avanti dell'ESR, Equivalent-Series-Resistance), gli interruttori abbiano resistenza zero quando chiusi e resistenza infinita quando aperti, senza rimbalzi.

Supponiamo di avere un carico di 12 Ω collegato all'uscita, e inizialmente $V_{OUT} = V_{IN} = 12V$, essendo l'interruttore 2 chiuso. Per la legge di Ohm, la corrente che scorre nel carico, attraversando sia l'induttanza che l'interruttore 2, è pari a $I_L = 1A$. All'istante t = 0, chiudiamo l'interruttore 1 e apriamo l'interruttore 2 (la linea tratteggiata in figura sta a significare che il comando è simultaneo). Come sappiamo, la corrente I_L nell'induttanza inizierà a crescere linearmente (fase 1). Ad un certo istante t1, apriamo l'interruttore 1 e chiudiamo l'interruttore 2: la corrente nell'induttanza inizierà a decrescere (non può certo continuare a crescere all'infinito). Dalla formula 2, una corrente che decresce ($\Delta I_L < 0$), implica necessariamente $V_L < 0$, perché sia Δt che L sono sempre positivi. **In pratica ai capi dell'induttanza si avrà un'improvvisa inversione di polarità.** Questa è la classica "flyback action" di tutte le induttanze… chiaro a cosa serve il diodo in antiparallelo alla bobina di tutti i relé? (ad evitare che all'apertura del relé, l'improvvisa inversione di polarità danneggi il transistor di pilotaggio). Osservando il circuito nella fase 2, si nota che la tensione di uscita è pari alla somma della tensione di ingresso e di quella "generata" dall'induttanza, cioè in definitiva, una tensione maggiore di quella di ingresso. Il condensatore C ha il compito di livellare la tensione di uscita che sarebbe altrimenti impulsiva, come quella in uscita da un normale diodo rettificatore connesso ad un trasformatore di rete. Questo ciclo viene ripetuto migliaia di volte al secondo. In figura 5 è riportato il grafico della corrente nell'induttanza.

In condizioni esterne stabili (cioè con tensioni di ingresso e di uscita stabili e resistenza di carico fissa) la corrente nell'induttanza oscilla sempre tra due valori, minimo e massimo, ben definiti e fissi, come nella figura 5. Se questi limiti variassero, nonostante le condizioni esterne stabili, il nostro switching avrebbe qualche serio problema (cortocircuito, instabilità del sistema di controllo, ecc.). La costanza di questi limiti implica che ΔI_L durante la fase 1 di crescita della corrente, deve essere uguale (in valore assoluto) a ΔI_L durante la fase 2 di diminuzione della corrente:

$$|\Delta I_{L,FASE1}| = |\Delta I_{L,FASE2}|$$

cioè, utilizzando ancora la prima formula e considerando che L è costante:

$$|V_{L,FASE1}| * t_{FASE1} = |V_{L,FASE2}| * t_{FASE2}$$

Possiamo chiamare la fase 1 come fase ON, in quanto è la fase in cui è chiuso l'interruttore 1; corrispondentemente chiameremo la fase 2, fase OFF. Riscrivendo l'equazione precedente:

Formula 4
$|V_{L,ON}| * t_{ON} = |V_{L,OFF}| * t_{OFF}$
(V_L in Volt, tensione ai capi di L)

Questa è una fondamentale formula che, sebbene derivata in questo caso particolare, **vale per tutte le induttanze e per qualunque tipologia di switching.**

INTERRUTTORI NEGLI ALIMENTATORI SWITCHING

Nella pratica, cosa si usa come interruttore? La risposta è molteplice, infatti si possono utilizzare: diodi (tradizionali e schottky), SCR, transistor bipolari BJT (NPN e PNP), MOSFET, IGBT, dipendentemente dalla posizione nel circuito (e quindi dalla tipologia) e dalle caratteristiche dello switching. Una precisazione va subito fatta: i diodi non sono controllabili (essendo a due soli piedini), quindi non possono essere utilizzati in tutte le posizioni del circuito. Per esempio, non possiamo utilizzare un diodo nella posizione 1 del circuito di figura 4, in quanto l'induttanza sarebbe sempre collegata alla tensione di ingresso, la corrente aumenterebbe costantemente con conseguente rottura del più debole tra induttanza, alimentatore di ingresso e diodo. Potremmo però tranquillamente utilizzarlo nella posizione 2, perché durante il periodo ON, quando è chiuso l'interruttore 1, l'anodo è a massa (e quindi il diodo non conduce comportandosi quasi esattamente come un circuito aperto), mentre durante il periodo OFF, quando la sovratensione dell'induttanza si mette in serie alla tensione di ingresso, fa passare la corrente verso l'uscita (comportandosi approssimativamente come un circuito chiuso). Il diodo nella posizione 2 è realmente utilizzato nella stragrande maggioranza dei casi. Nella posizione 1 occorre un interruttore comandato, e la scelta è fra SCR, transistor a giunzione BJT, MOSFET e IGBT. Tralasciando gli SCR, ormai non più utilizzati negli switching tradizionali, i tre dispositivi rimanenti si dividono il numero di applicazioni, con una forte prevalenza dei MOSFET. I transistor BJT resistono in applicazioni ad alta tensione dove possono risultare i più economici a parità di prestazioni, ed anche in applicazioni a bassa tensione dove non è disponibile "l'alta" tensione di pilotaggio richiesta dai MOSFET. Gli IGBT sono di preferenza utilizzati per potenze elevate, dell'ordine dei kW o più. Per i nostri circuiti useremo quindi MOSFET e BJT. Una precisazione importante: un interruttore **ideale** ha solo due stati stabili: ON oppure OFF. Nello stato ON la tensione ai suoi capi vale zero, mentre la corrente può assumere qualsiasi valore. La potenza dissipata dall'interruttore in questo

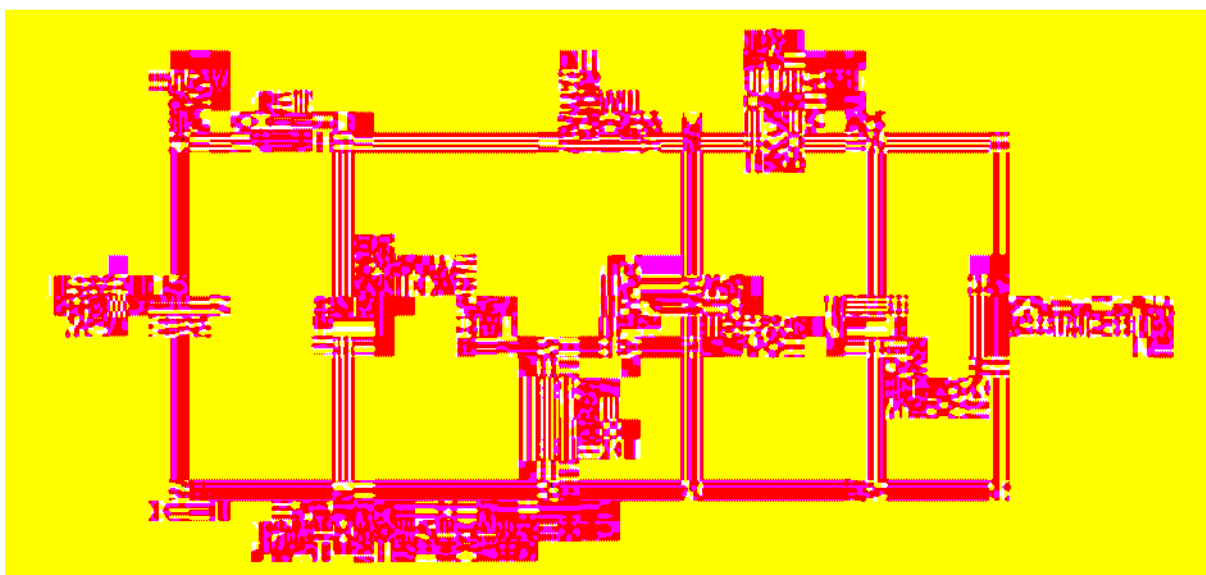



Figura 6: *Progetto boost – parte di potenza.*

stato è quindi:

$P_{DISS,ON} = V_{ON} * I_{ON} = 0 * I_{ON} = \mathbf{0}$

Analogamente, nello stato OFF, la tensione ai capi dell'interruttore può essere qualsiasi ma la corrente che scorre è zero:

$P_{DISS,OFF} = V_{OFF} * I_{OFF} = V_{OFF} * 0 = \mathbf{0}$

Un MOSFET e un BJT non sono un interruttore ideale, ma lo approssimano abbastanza bene se fatti lavorare in saturazione ed in interdizione, avendo cura di evitare la zona attiva, dove sia V che I sono diversi da zero, e quindi la dissipazione è alta. Questa "accortezza" è di importanza fondamentale per realizzare switching ad alta efficienza.

IL PROGETTO, FINALMENTE

Vediamo ora di mettere in pratica le poche nozioni (sono solo quattro formulette!) che ho presentato.

Realizzeremo un alimentatore di tipo boost, quindi con $V_{OUT} > V_{IN}$, senza alcun circuito di controllo della tensione di uscita.

Nonostante questo, e a patto di soddisfare alcune condizioni, **la tensione di uscita è molto stabile al variare del carico!** Non è stabile al variare della tensione di ingresso, anzi in realtà questo circuito è un moltiplicatore x2 della tensione di ingresso. Per questo primo progetto i parametri saranno a tolleranza zero, cioè non considereremo variazioni dei valori dei componenti, per non confondere troppo le idee. Gli effetti delle tolleranze dei componenti sui calcoli saranno affrontati successivamente. Lo schema della parte di commutazione (o "di potenza") del nostro primo switching è riportata in figura 6.

Ho usato un MOSFET a canale N di tipo IRF620 come interruttore comandato 1, ed un diodo Schottky 1N5818 come interruttore automatico 2.

È ovvio che potete utilizzare qualsiasi altro componente di caratteristiche simili.

Esempi di alternative per il MOSFET: IRF510, BUZ73, BUZ32, IRF640, IRF540, in pratica qualunque NMOS con corrente max. di drain uguale o superiore a 4A e tensione max. di drain uguale o superiore a 30V.

Per il diodo: 1N5819, 1N5821 (no 1N5820 e 1N5817 perché sopportano solo 20V di tensione inversa), MBR160, SS14 (SMD).

La resistenza R1 ha il compito di tenere spento il MOSFET qualora non fosse collegato il circuito di pilotaggio presentato nelle pagine successive. Il condensatore C1 diminuisce il ripple di corrente richiesto all'ingresso, mentre il ponticello JP1 permette di misurare la corrente media assorbita collegando al suo posto un amperometro.

Parametri di progetto desiderati:

$V_{IN} = +12V$

$V_{OUT} = +24V$

$I_{OUT} = 0{,}12A$ max.

Possiamo definire il metodo di calcolo usato, "bilancio di potenza iterativo".

PASSO 1 - calcolo della massima potenza di uscita

$P_{OUT,MAX} = V_{OUT} * I_{OUT,MAX}$

Nel nostro caso:

$P_{OUT,MAX} = 24 * 0{,}12 = 2{,}88W$

PASSO 2 - stima dell'efficienza e calcolo della massima potenza di ingresso

È una stima perché non sappiamo a priori quanto sarà efficiente il nostro switching.

Sceglieremo quindi un valore a piacere compreso tra il 70 e 90%, calcoleremo tutti i parametri e alla fine verificheremo se la scelta iniziale era corretta: in caso contrario ripeteremo il calcolo con il nuovo valore di η. Di qui il nome del procedimento iterativo.

Tanto per cominciare, è evidente che la potenza assorbita in ingresso sarà sicuramente maggiore di quella fornita in uscita.

Ricordate la formula 1?

$\eta = P_{OUT} / P_{IN} * 100$

da questa si ricava:

$P_{IN} = P_{OUT} / \eta * 100$

Scegliendo η = 80%, otteniamo:

$P_{IN,MAX} = 2{,}88 / 80 * 100 = 3{,}6W$

PASSO 3 - calcolo della massima corrente di ingresso

Semplice:

$I_{IN,MAX} = P_{IN,MAX} / V_{IN}$

Nel nostro caso:

$I_{IN,MAX} = 3{,}6 / 12 = 0{,}3A = I_{AVG}$

Notate che nella tipologia boost questa è la stessa corrente che scorre nell'induttanza.

Un momento: abbiamo detto che la corrente nell'induttanza sale e scende nelle due fasi ON e OFF!

Tranquilli, quella che abbiamo calcolato è solo la corrente media I_{AVG} che viene assorbita dall'ingresso (ad essere precisi, è la corrente RMS, ma la differenza in questo caso è trascurabile, l'errore è inferiore al 4%).

PASSO 4 - calcolo del duty-cycle a regime

Abbiamo già detto che il nostro

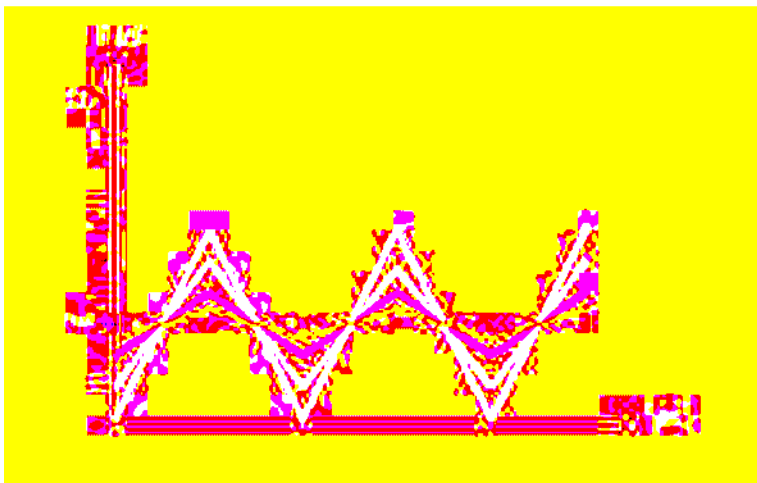



Figura 7: Corrente nell'induttore, intorno al valore medio 0,3A, ripple variabile.

switching funziona a due fasi, riferite all'interruttore 1: ON e OFF. Calcoliamo il duty-cycle a regime, cioè il periodo ON rispetto al totale del ciclo (ON+OFF), usando l'importantissima formula 4:

$|V_{L,ON}| * t_{ON} = |V_{L,OFF}| * t_{OFF}$

$V_{L,ON}$ è la tensione ai capi dell'induttanza durante la fase ON. Se supponiamo che l'interruttore 1 sia perfetto, $V_{L,ON}$ = 12V. $V_{L,OFF}$ è la tensione ai capi dell'induttanza durante la fase OFF. Se supponiamo che l'interruttore 2 sia perfetto, $V_{L,OFF}$ = 24 – 12 = 12V. Quindi:

$12 * t_{ON} = 12 * t_{OFF}$

cioè

$t_{ON} = t_{OFF}$

e quindi

duty-cycle = $t_{ON}/(t_{OFF}+t_{ON})$ = 50%

PASSO 5 - scelta della frequenza di commutazione e dell'induttanza

La frequenza di commutazione è praticamente a piacere, in quanto esistono infinite coppie induttanza-frequenza che forniscono in prima approssimazione lo stesso risultato. Si parte da un minimo assoluto di 15kHz, fino ad un massimo per il nostro corso, di 100-200kHz. Sotto ai 15kHz sono necessarie induttanze molto grandi e c'è il rischio di un "fischio" udibile dall'alimentatore.

Sopra ai 200kHz entrano in gioco in maniera evidente le perdite magnetiche, l'effetto pelle nei conduttori in rame e altri fattori.

Dal calcolo del duty-cyle abbiamo, per la corrente attraverso l'induttore, un grafico simmetrico (per il 50% del tempo la corrente sale, per il rimanente 50% scende), con centro intorno a 0,3A, che è la corrente media assorbita dall'ingresso. Di grafici con queste caratteristiche ne esistono però infiniti, vedi figura 7. Ciò che cambia tra queste curve, tutte centrate intorno alla corrente media, è l'ampiezza del "ripple" di corrente picco-picco. Questo determina il cosiddetto carico critico o minimo dell'alimentatore, sotto al quale l'alimentatore è instabile o meglio lavora in modo discontinuo. Nel nostro caso non verrebbe più garantita la tensione di uscita stabile a +24V. La spiegazione è la seguente: supponiamo di scegliere il grafico in rosso nella figura 7. In questo caso il ripple è pari al 200% del valore medio, e cioè 0,6A. Se la corrente assorbita dal carico fosse esattamente pari al massimo, lo switching si troverebbe in una situazione di stabilità. Supponiamo ora che la corrente di carico si dimezzi.

Teoricamente anche la corrente media nell'induttore dovrebbe scendere alla metà di 0,3A = 0,15A, **con sovrapposto lo stesso ripple di 0,6A in quanto dalla Formula 2, se la frequen-**

Figura 8: Corrente nell'induttore con ripple = 50%, a carico max. e min.

za di switching e L sono costanti, il ripple è costante. In teoria quindi, dal grafico la corrente dovrebbe scendere sotto lo zero, ma questo è chiaramente impossibile in quanto non può scorrere corrente verso l'alimentatore/batteria di ingresso (c'è il diodo). Accade invece che la tensione di uscita aumenterà, per far si che anche la corrente nel carico aumenti, e la corrente media nell'induttanza arrivi a 0,3A. Le "piacevoli" conseguenze per il carico le lascio alla vostra immaginazione. Questo è ciò che accade realmente al nostro primo circuito, senza controllo della tensione di uscita: senza carico la tensione sale fino a 50-60V!
Per questo motivo, un ripple pari al massimo (200%) non è praticamente utilizzabile. In genere conviene scegliere un ripple picco-picco pari a circa il **20-50% della corrente media nell'induttore**. Scegliamo il 50% di ripple, cioè 0,15A pk-pk (50% di 0,3A).
A massimo carico, la corrente massima nell'induttore sarà pari a 0,3 + 0,15 / 2 = 0,375A mentre la corrente minima sarà 0,3 - 0,15 / 2 = 0,225A, come in figura 8.

Da questa figura è immediato calcolare la specifica di corrente minima di uscita per l'alimentatore. Questa sarà pari alla corrente massima di uscita, per il ripple appena scelto in %, diviso 2:
$I_{OUT,MIN}$ = 0,12 * 50% / 2 = 0,03A

Proseguiamo con i calcoli, ricordando la formula 2:
$\Delta I_L = (V_L / L) * \Delta t$
e applicandola alla sola fase ON (si può anche applicare alla sola fase OFF, non cambia nulla in quanto sappiamo che ΔI_L nelle due fasi deve essere uguale ed opposto):
$t_{ON} = L * \Delta I_L / V_L = L * 0,15 / 12 =$
$= L * 0,0125$

Come ho anticipato, per l'induttanza si deve effettuare un compromesso. Induttanze di piccolo valore si "caricano" velocemente e quindi richiedono una frequenza di commutazione più alta e viceversa. Conviene utilizzare valori standard per L e tabulare le frequenze che ne derivano (vedi tabella seguente).

Induttanza L in µH	Valore di t_{ON} in µs	Freq. switching f_{SW} = (duty / 100) / t_{ON}
220	2,750	182kHz
470	5,875	85kHz
1000	12,500	40kHz

Le induttanze Panasonic della serie ELC08 (reperibili sul catalogo RS) sopportano le seguenti correnti massime:

Induttanza L in µH	Corrente massima in A
220	0,6
470	0,4
1000	0,3

L'induttanza da 1000µH è sottodimensionata, la 470µH è al limite, la 220µH è accettabile come corrente massima sebbene richieda una frequenza di switching abbastanza elevata.
Scegliamo quindi la 220µH, fatta lavorare alla frequenza di circa 180kHz.

PASSO 6 - stima (o misura) delle perdite dissipative
Per mancanza di spazio presenterò il metodo di calcolo nella prossima puntata, per ora accontentatevi dei risultati pronti.
PERDITE NEL DIODO, NEL MOSFET E NELL'INDUTTANZA: 0,16W

PASSO 7 – calcolo dell'efficienza e verifica della bontà della stima
È evidente che la potenza assorbita dall'ingresso è pari alla somma della potenza erogata e di quella dissipata:
$P_{IN} = P_{OUT} + P_{DISS}$
Quindi, dalla formula 1:
$\eta = P_{OUT} / P_{IN} * 100 = 2,88 / (2,88 + 0,16) * 100 = 94,7\%$
L'efficienza calcolata è diversa (maggiore) di quella ipotizzata. Si possono quindi reiterare i calcoli, partendo dal PASSO 2, e utilizzando questo nuovo valore per l'efficienza. **Tuttavia il circuito è perfettamente funzionante anche mantenendo conservativamente l'efficienza al 80%**. La sola differenza è che la scarsa ottimizzazione potrebbe portare alla scelta di componenti sovradimensionati. In caso contrario, cioè se l'efficienza calcolata fosse inferiore a quella stimata, sarebbe **obbligatorio**

ripetere i calcoli.

PASSO 8 - calcolo del condensatore di uscita (preliminare)

Nella tipologia boost, abbiamo già notato che il condensatore di uscita viene caricato solo durante il periodo t_{OFF} e si scarica sul carico nel periodo t_{ON}.
Non esiste quindi un flusso continuo di corrente dall'ingresso all'uscita, (a differenza di altre tipologie) e occorre un condensatore di valore maggiore, a parità di altre caratteristiche. Il valore dipenderà dal carico e dall'ammontare di ripple della tensione di uscita tollerabile (da non confondere con il ripple di corrente nell'induttanza).
Possiamo considerare un ripple pari a 100mV picco-picco come massimo tollerabile per le apparecchiature elettroniche.
Tuttavia, se avete necessità di alimentare ad esempio un relé, potete tranquillamente aumentare questo valore anche a 1-2Volt e risparmiare sulle dimensioni e sul costo del condensatore.
La quantità di carica Q immagazzinata in un condensatore C, carico a tensione V, vale:
$Q = C * V$
(Q in Coulomb, C in Farad, V in Volt).
Se si preleva o si fornisce una certa quantità di carica dal condensatore (cioè si assorbe o si fornisce corrente), la tensione ai suoi capi varierà di:
$\Delta V = \Delta Q / C$
La quantità di carica prelevata o fornita ΔQ è pari alla corrente I per il tempo Δt che dura il prelievo o la fornitura:
$\Delta Q = I * \Delta t$
Dalle precedenti equazioni si ottiene:
$\Delta V = I * \Delta t / C$, o meglio:

Formula 5
$C = I * \Delta t / \Delta V$
(C in Farad, I in Ampere, Δt in secondi, ΔV in Volt)

In tutte le tipologie, il calcolo del condensatore di uscita si fa generalmente analizzandone il comportamento nella fase ope-

Figura 9: Progetto boost - circuito di pilotaggio, duty-cycle = 50%.

rativa più comoda, in quanto è ovvio che il ripple di tensione è identico e opposto nelle due fasi ON e OFF (se così non fosse la tensione di uscita salirebbe all'infinito o scenderebbe a zero). Nella tipologia boost conviene utilizzare la fase ON, durante la quale l'interruttore 2 (diodo D1) è aperto, e quindi il condensatore si scarica sul carico. Sappiamo già che il tempo t_{ON} è pari a 2,75µs, sostituendo ΔV = 0,1V e I = 0,12A si ottiene dalla formula 5:

$$C = 0{,}12*2{,}75*10^{-6}/0{,}1 = 3{,}3\mu F$$

NOTA: poiché l'avevo disponibile, ho utilizzato un condensatore elettrolitico da 10µF, del tipo LOW-ESR/105°. Non utilizzare condensatori elettrolitici tradizionali (85°)! Mi aspetto in questo caso un ripple ridotto a un terzo (in quanto ho triplicato la capacità), cioè circa 30mV picco-picco.

CIRCUITO DI PILOTAGGIO

Il circuito di pilotaggio, visibile in figura 9, è basato sul noto 555, in versione CMOS e configurazione astabile. Avendo necessità di un duty-cycle pari al 50%, ho utilizzato la configurazione in figura invece di quella più classica con due resistenze R_A e R_B. Per aumentare la corrente di uscita (ne vedremo il motivo nelle prossime puntate), ho usato un push-pull inseguitore di emettitore a transistor complementari di tipo BC337 o BC327.

Al solito, può essere utilizzata qualunque coppia di transistor

Corrente assorbita a Vin = 12V [A]	Potenza assorbita [W]	Resistenza di carico [Ω]	Tensione di uscita [V]	Potenza fornita [W]	Efficienza η [%]
0,242	**2,904**	200	23,46	**2,752**	94,8
0,163	**1,956**	300	23,77	**1,883**	96,3
0,123	**1,476**	400	23,92	**1,430**	96,9
0,071	**0,852**	700	24,06	**0,827**	97,1
0,065	**0,780**	800	24,53	**0,752**	96,4
0,065	**0,780**	900	26,02	**0,752**	96,4
0,068	**0,816**	1000	28,10	**0,790**	96,8

come 2N2222 o 2N2904, BC847 o BC857, eccetera. Il trimmer RV1 permette la regolazione della frequenza di commutazione entro ampi margini per dare la possibilità di provare vari circuiti.
Per il montaggio dei circuiti è sufficiente una basetta millefori, a patto di ridurre al minimo la lunghezza dei collegamenti tra induttanza, MOSFET, diodo e condensatori. Il circuito di pilotaggio non è assolutamente critico. Prima dell'accensione, ricordate di collegare in uscita un carico, ad esempio una resistenza di 2-3W, di valore pari o inferiore a:

$R_{MIN} = V_{OUT} / I_{OUT,MIN} = 24 / 0,03 = 800\ \Omega$

MISURE SUL CIRCUITO

L'efficienza misurata a massimo carico è risultata pari al 95%. **Ricordate che, ai fini dei calcoli di progetto, l'efficienza va misurata considerando solo la parte di potenza, e non il circuito di pilotaggio** (usare l'amperometro in JP1 di figura 6). Le prove con carico variabile hanno fornito i risultati riportati nella tabella in alto.

Notate come nelle ultime due righe, per carichi inferiori al minimo (resistenza maggiore di 800Ω), la tensione di uscita salga, come previsto, ben oltre i 24V nominali. Se volete provare carichi ancora minori, fate attenzione a non superare la tensione massima sopportabile dal condensatore di uscita, dal diodo e/o dal mosfet.
Per concludere in figura 10 ho riportato il ripple della tensione di uscita (ovviamente con l'oscilloscopio in AC).
Ma il ripple picco-picco è di circa 750mV invece dei 30mV previsti!
Per la spiegazione vi rimando alla prossima puntata, fate pure le vostre supposizioni.

CONCLUSIONE

Concludo esortandovi a provare altri circuiti di tipo boost, variando la frequenza di switching, l'induttanza e la tensione di ingresso, rimanendo per quest'ultima nel range 8-18V (attenzione che alcune versioni CMOS del 555 sopportano al massimo 15V).
La prossima puntata sarà dedicata al calcolo delle perdite negli alimentatori switching, e ad un semplice circuito di controllo per il boost appena realizzato.

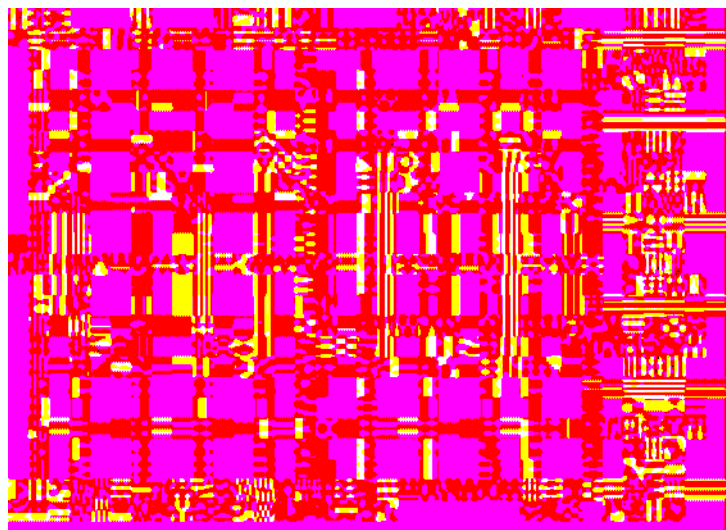



Figura 10: *Progetto boost – ripple di tensione di uscita.*

FREQUENZIMETRO DIGITALE BF 4,5 DIGIT LED

di Umberto Fabris
oneberto@libero.it

Il progetto che mi accingo a presentare è di facile realizzazione e contribuirà ad arricchire il vostro laboratorio. Si tratta di un sensibile e preciso frequenzimetro digitale per BF controllato da microprocessore.

L'obiettivo di questo progetto è la realizzazione di uno strumento in grado di leggere, con elevata precisione, la frequenza di un segnale nel campo delle applicazioni BF, che deve, inoltre, possedere una sensibilità notevole, capace di adattarsi anche a segnali di rilevante ampiezza. Una elevata impedenza di ingresso e la possibilità di essere alimentato con qualunque sorgente di tensione, continua o alternata, ne aumentano la versatilità (da 6 a 24 Vac e da 8 a 35 Vdc). Da non sottovalutare, poi, la possibilità di leggere a distanze di almeno un paio di metri, in ambienti fortemente illuminati, il valore di frequenza indicato.

Quello che ne è nato è uno strumento dotato di una impedenza di ingresso di 1MΩ, capace di accettare segnali nel range compreso da 20 mV a 100 Vpp, con una visualizzazione su display LED a 7 segmenti, in grado di misurare su due portate con tempo di gate di 1 Sec. e 100 mS, frequenze fino a 200 Khz, con risoluzioni, rispettivamente, di 1 Hz e 10 Hz.

La precisione di lettura è garantita dalla elevata frequenza del quarzo che cadenza il funzionamento del microcontrollore usato per il controllo delle temporizzazioni: 20 MHz; da cui otteniamo, attraverso un appropriato software, il preciso impulso di gate, selezionabile tra 1 Sec e 100 mS e gli impulsi di strobe e di reset, necessari al corretto funzionamento del circuito integrato contatore e pilota display ICM7225.

Nell'esemplare realizzato, il range massimo di lettura è stato fissato, per mia scelta, a 200 KHz; programmando il

microcontrollore, con un'altra versione di software, per generare impulsi di gate di 100 mS e 10 mS, si possono misurare segnali fino a 2 MHz con risoluzione di 100 Hz.

IL CIRCUITO

Come possiamo vedere dalla figura 1 (che mostra il circuito di ingresso, di alimentazione e di temporizzazione), il segnale da misurare viene applicato al gate del FET FT1 per ottenere un' alta impedenza di ingresso. I segnali di ampiezza elevata, vengono tosati per mezzo dei diodi D1÷D4 e della resistenza R1. Il segnale a bassa impedenza, dis-

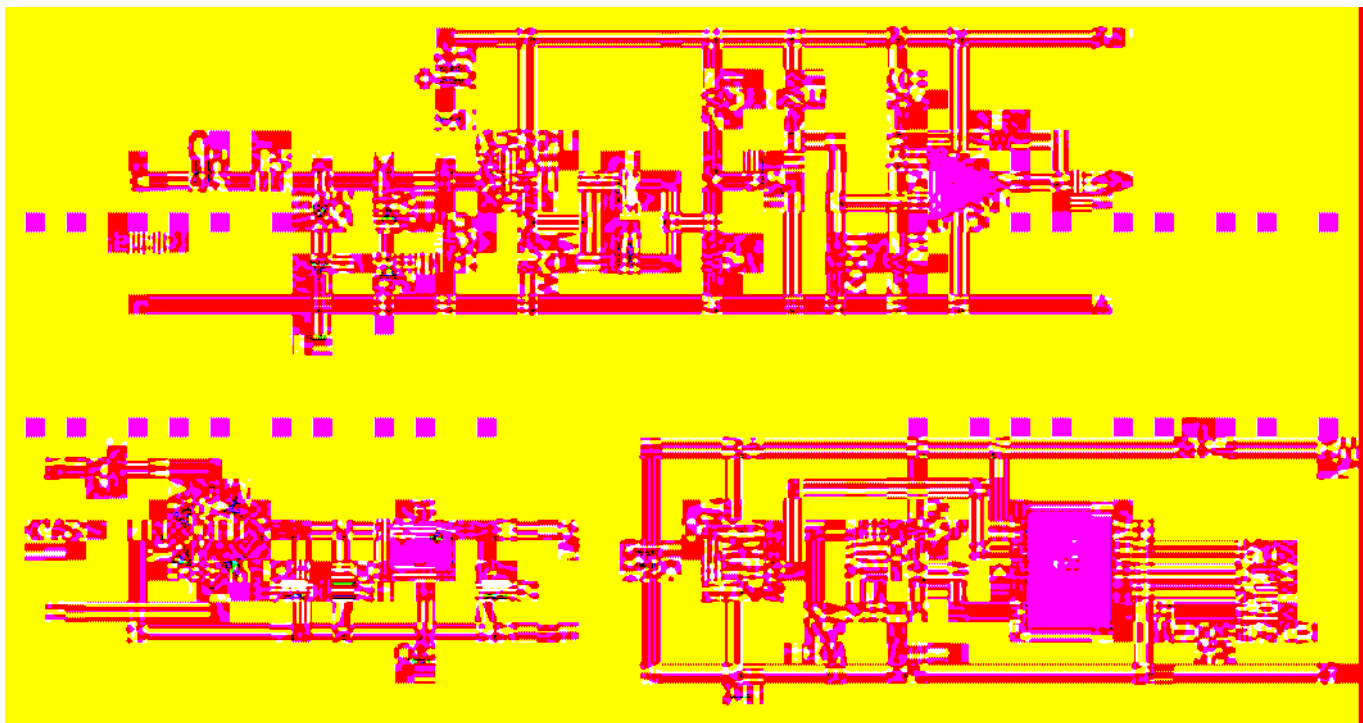



Figura 1: *Schema elettrico della scheda principale*

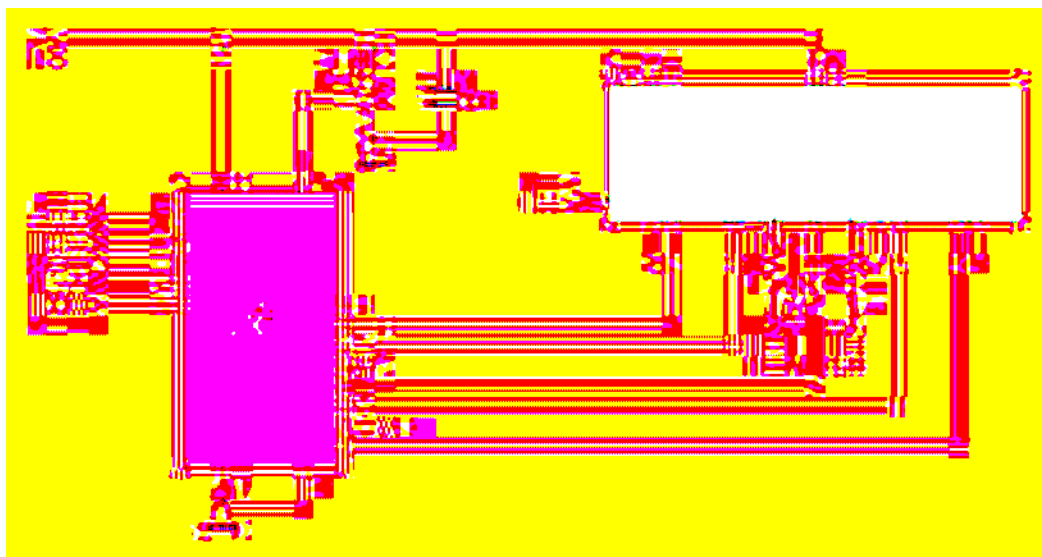



Figura 2: *Schema elettrico della scheda di visualizzazione*

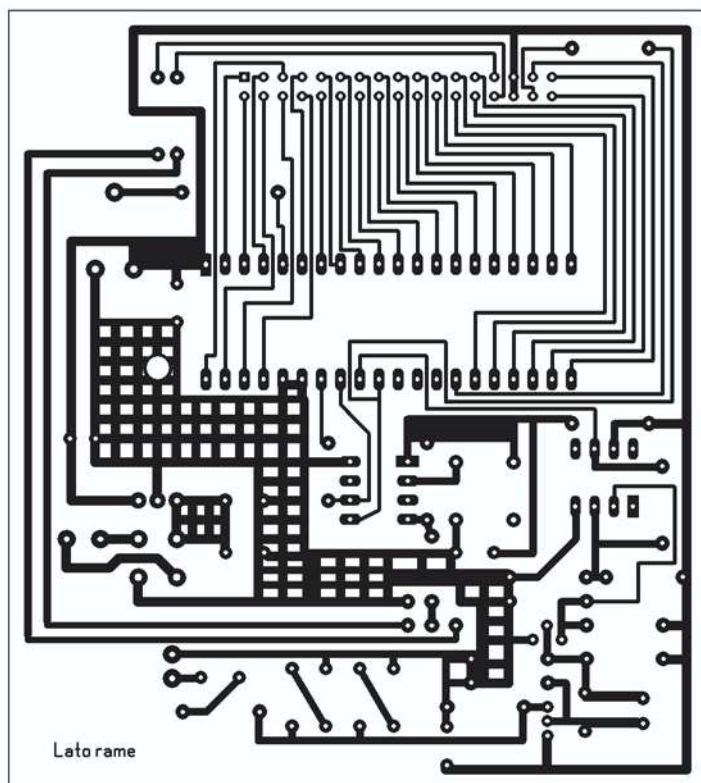



Figura 3: Circuito stampato scala 1:1 (lato rame) della scheda principale

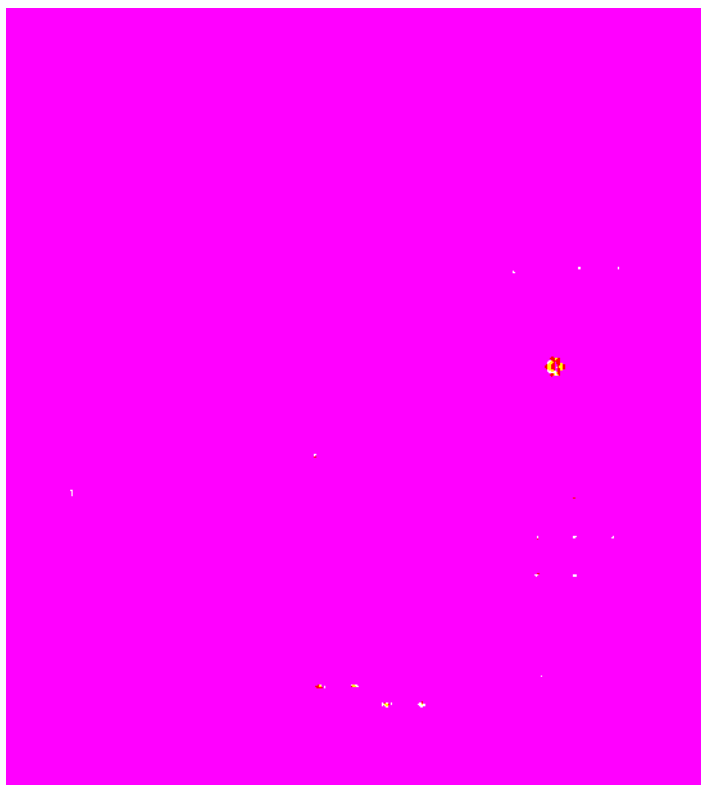



Figura 4: Piano di montaggio componenti della scheda principale

ponibile ai capi di R3, viene successivamente amplificato dal transistor TR1 ed inviato all'ingresso invertente del circuito integrato amplificatore operazionale IC3.
Questo operazionale viene usato come "trigger di Schmitt", con soglia di scatto superiore a 2 V ed inferiore a 1 V, presentando quindi alla sua uscita, indipendentemente dalla forma del segnale al suo ingresso, degli impulsi ad onda quadra con fianchi ripidi, adatti a pilotare correttamente l'ingresso di conteggio al pin 32 dell'integrato ICM7225.
Questo integrato, come si è detto, necessita oltre che di un accurato impulso di gate necessario ad abilitare il conteggio, di altri due impulsi: l'impulso di strobe che memorizza la lettura sul display e l'impulso di reset che azzera il contatore interno e lo prepara ad una nuova lettura.
Tutte queste funzioni vengono gestite dal software memorizzato all'interno di IC2, un PIC12F675 oppure PIC12F629 (i due modelli si distinguono per la presenza o meno della funzione ADC, peraltro non utilizzata in questo progetto).
Il software “sente” il livello alto o basso selezionato dallo switch SW2 sul pin 4 e cambia la durata dell'impulso di gate al pin 5 selezionandola a 1 Sec o a 100 mS. Successivamente genera in sequenza al pin 6 l'impulso di strobe e al pin 7 quello di reset.
Nello schema riportato in figura 2, possiamo vedere il semplice stadio di conteggio e di visualizzazione effettuato dall’ICM7225, in grado come si è detto, di effet-

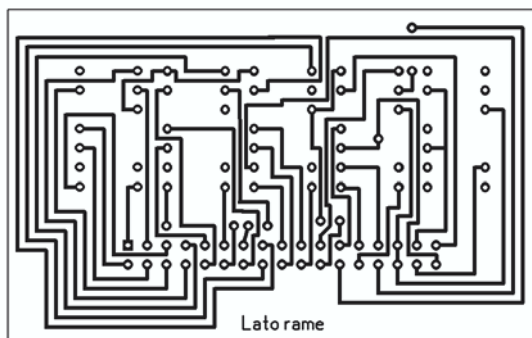



Figura 5: *Circuito stampato scala 1:1 (lato rame) della scheda di visualizzazione*

Figura 6: *Piano di montaggio componenti della scheda di visualizzazione*

tuare il conteggio degli impulsi applicati al pin 32 e di pilotare i 5 display ad anodo comune. Una particolarità di questo integrato è l'assenza di sfarfallio delle cifre in quanto il pilotaggio del display è diretto e non multiplexato, inoltre, tramite il trimmer VR1 è possibile regolare la luminosità degli stessi in base all'illuminazione ambientale ed ai nostri gusti.

LA COSTRUZIONE

Per permettere a chiunque di realizzare il frequenzimetro ho deciso di fornire il firmware necessario a programmare il PIC utilizzato nel progetto, lo potete quindi scaricare dal sito di Fare Elettronica.

Il firmware è disponibile in 2 versioni:

- Massima frequenza di lettura

Elenco componenti

Sigla	Valore	Sigla	Valore
R1	27 KΩ 1/4 W	C5	220 KpF 100 V poliestere
R2	1 MΩ 1/4 W	C7	220 KpF poliestere
R3	680 Ω 1/4 W	C8	4,7 µF 16 V tantalio
R4, R9, R10	10 KΩ 1/4 W	L1	47 uH assiale
R5	47 KΩ 1/4 W	TR1	BC547C
R6	1 KΩ 1/4 W	FT1	2SK218 - BF245
R7	100 KΩ 1/4 W	PT1	Ponte 1 A 200 V
R8	3,3 KΩ 1/4 W	D1÷4	1N4148
R11	18 KΩ 1/4 W	Display	5 display 7 segmenti - Anodo comune - doppio punto decimale
R12÷R14	330 Ω 1/4 W	IC1	7805
VR1	Trimmer 10 KΩ 1/4 W orizzontale	IC2	PIC16F675/629
C1	1000 µF 35 V elettrolitico	IC3	LF357
C2	1 µF 16 V elettrolitico	IC4	ICM7225
C3	220 µF 16V elettrolitico	SW1	Interruttore a levetta
C4, C6, C9	100 KpF poliestere	SW2	Doppio deviatore a levetta

di 200 KHz (risoluzione 10 Hz);
- Massima frequenza di lettura 2 MHz (risoluzione 100 Hz).

Quindi, una volta reperiti i vari componenti, si può procedere alla realizzazione del circuito stampato principale, come proposto in figura 3 e 4 che alloggia tutti i componenti del frequenzimetro.
Il circuito stampato ospitante i 5 display, è invece visibile in figura 5 e 6.
I due circuiti verranno connessi attraverso un cavo piatto da 34 conduttori e relativi connettori (vedi figura 7).
Il circuito, una volta ultimato il montaggio dei componenti, dei commutatori, del cavo di interconnessione e del connettore BNC di ingresso, risulterà come visibile in figura 7.
A questo punto, se tutto è stato montato correttamente il frequenzimetro deve funzionare, presentando in assenza del segnale d'ingresso tutte le cifre spente, salvo vedere lampeggiare il punto decimale della cifra a sinistra ed il punto decimale fisso che indica i KHz di lettura. Spostando il deviatore SW2 il punto dei KHz si sposterà ed il punto del gate lampeggerà a diversa frequenza.
Provate ora ad iniettare un segnale all'ingresso e dovrete leggerne la frequenza esatta; il circuito è talmente sensibile che nell'esemplare da me realizzato, toccando con una mano l'ingresso e con l'altra la massa, riesce a leggere i 50 Hz della rete elettrica. Infine regolate il trimmer VR1 per la luminosità del display. Il circuito ora è pronto per essere inscatolato (figura 8) in un adeguato contenitore ed essere utilizzato nel vostro laboratorio.

Figura 7: *Foto del frequenzimetro montato e cablato*

Figura 8: *Foto del frequenzimetro inserito nel contenitore*

CONCLUSIONI

Spero che questo mio semplice progetto vi ritorni utile nel vostro laboratorio, la spesa per realizzarlo è modesta e la soddisfazione di vederlo funzionare tanta.

CAMPAGNA ABBONAMENTI 2004-2005

Abbonarsi a Fare Elettronica significa ricevere ogni mese, direttamente a casa, tante idee, consigli e progetti di assoluta novità.

Fare Elettronica trasforma il tuo lavoro o il tuo hobby in una vera passione.

Abbonandoti potrai ricevere la rivista ad un prezzo molto conveniente rispetto a quello di copertina. Pagherai infatti solo **€ 39,00** invece di € 51,00, con **un risparmio di ben € 12,00** e la sicurezza di un prezzo bloccato per un anno.

Inoltre riceverai un **buono sconto del 20%** utilizzabile per il tuo prossimo acquisto su www.farelettronica.com e, con la merce ordinata, riceverai il bellissimo portapenne di Fare Elettronica

ABBONARSI A FARE ELETTRONICA CONVIENE.

Abbonarsi subito conviene ancora di più.

Diverse possibilità di abbonamento:

Standard: Il tuo abbonamento personale o aziendale al costo di € 39,00

Regalo: Se sei già abbonato e vuoi regalare un nuovo abbonamento ad un amico, lo pagherai solo € 35,00 (10% di sconto) comunicando il tuo codice

Scuole: Riservato a scuole ed università; ordinando quattro abbonamenti ne riceverai uno in omaggio, pagherai quindi € 156,00 anzichè € 195,00 (20% di sconto)

Come abbonarsi:

Per Posta: scrivere a INWARE Edizioni - Via Cadorna, 27 - 20032 Cormano (MI)

Per Telefono: al numero +39 02.66504794

Per Fax: al numero +39 02.66508225

Via Internet: sul sito www.farelettronica.com alla pagina campagna abbonamenti

(è possibile pagare con bollettino postale, bonifico bancario e carta di credito, maggiori dettagli in ultima pagina)

FARE ELETTRONICA BUNDLES È UNA COMBINAZIONE DI PRODOTTI AD UN PREZZO SCONTATISSIMO.

NON LASCIARTI SCAPPARE IL TUO PREFERITO!

ECCO L'ELENCO AGGIORNATO

Elenco dei prodotti disponibili nei Bundles	Prezzo normale Cad.
❑ Fare Elettronica N. 220/221 Ottobre/Novembre 2003 – Numero Doppio	€ 9,00
❑ Fare Elettronica N. 222 Dicembre 2003	€ 7,50
❑ Fare Elettronica N. 223 Gennaio 2004	€ 7,50
❑ Fare Elettronica N. 224 Febbraio 2004	€ 7,50
❑ Fare Elettronica N. 225 Marzo 2004	€ 7,50
❑ CD-ROM "Annata Fare Elettronica 2003"	€ 25,00
❑ Software Abacom "sPlan Disegno schemi elettrici"	€ 46,80
❑ Software Abacom "sPrint Disegno circuiti stampati"	€ 46,80
❑ Software Abacom "Front Designer Disegno pannelli frontali"	€ 46,80

Elenco dei Bundles disponibili	Prezzo Bunble Cad.
❑ Bundle "A" : 2 Arretrati (qualsiasi) al prezzo speciale di	€ 12,00
❑ Bundle "B" : 3 Arretrati (qualsiasi) al prezzo speciale di	€ 15,00
❑ Bundle "C" : 1 Arretrato (qualsiasi) e il CD-ROM "Annata FE 2003" al prezzo speciale di	€ 30,00
❑ Bundle "D": 1 Arretrato (qualsiasi) e 1 Software Abacom (qualsiasi) al prezzo speciale di	€ 50,00
❑ Bundle "E" : 2 Software Abacom (qualsiasi) al prezzo speciale di	€ 84,00

Indicare il Bundle e i prodotti prescelti mettendo una croce sulle rispettive caselle.

Tutti i Bundle sono comprensivi di IVA e spese di spedizione. Il pagamento è anticipato, e può essere effettuato tramite:

❑ **Bollettino postale** Utilizzare il **C/C N. 22790232** intestato ad **Inware srl**, indicando nella causale **"Abbonamento a Fare Elettronica"**

❑ **Bonifico bancario** Appoggiarlo sulla banca: **Poste Italiane - CIN: Z - ABI: 07601 - CAB: 01600 - C/C: 000022790232**

❑ **Carta di credito** Titolare: ________________
Numero: ________________ Scadenza: ___ / ___

VISA MasterCard

Cognome ________________ Nome ________________

Azienda ________________

Via ________________ CAP ________ Città ________________ Prov. ________

Tel. ________________ Fax ________________ email ________________

Privacy. Ai sensi dell'art. 13 D. lgs. 196/03, e successive modificazioni, Vi comunichiamo che il trattamento dei dati, in forma automatizzata e con modalità strettamente connesse ai fini, con garanzia di riservatezza, è finalizzata all'invio del presente periodico allo scopo di informare ed aggiornare i lettori e gli operatori del settore elettronico sulle novità che il mercato propone. Potranno essere esercitati i diritti di cui all'Art. 7 del D.lgs. 196/03 (accesso, correzione, cancellazione, opposizione al trattamento ecc.). Il titolare del trattamento dei dati è Inware srl con sede a Cormano (MI) in via Cadorna 27/31. Nel caso si tratti di copia omaggio a titolo promozionale si rende noto che i dati provengono da archivi pubblici. Resta inteso che le informazioni in ns. possesso non saranno in nessun caso cedute a terzi.

VITAMINA C
quattordicesima parte

IL COMPILATORE SDCC E L'8052

di Antonio Di Stefano
a.distefano@farelettronica.com

In questa puntata considereremo uno strumento molto interessante per la programmazione in C dei microcontrollori: il compilatore SDCC. Questo compilatore, oltre ad avere delle ottime capacità ha anche il pregio di essere completamente freeware. Oltre al compilatore verrà descritto anche il suo target predefinito: il celebre microcontrollore 8052, che come vedremo per le sue caratteristiche si presta particolarmente bene ad eseguire codice C.

INTRODUZIONE

La disponibilità di compilatori C per microcontrollori è un fatto piuttosto recente. Fino ad una decina di anni fa la tecnologia dei compilatori era meno raffinata di quella odierna e non consentiva di ottenere del codice particolarmente efficiente e compatto. Questo era dovuto in gran parte al fatto che lo sviluppo dei compilatori aveva sempre avuto come obbiettivo quello di generare codice per macchine piuttosto complesse, tipicamente basate su microprocessori e dotate di abbondanti risorse di memoria (i computer). Rispetto a questi sistemi i microcontrollori offrono risorse estremamente limitate, set di istruzioni molto meno flessibili, e velocità di esecuzione ridotta. Questa "incompatibilità" ha fatto si che i microcontrollori siano sempre stati tradizionalmente programmati in assembler per riuscire a sfruttarne al massimo le caratteristiche. Tuttavia i vantaggi di un linguaggio ad alto livello come il C sono innegabili: maggiore produttività, maggiore documentabilità e manutentibilità del codice, ecc. Questo ha spinto alla creazione di compilatori sempre più ottimizzati ed efficienti anche per piccoli microcontrollori.

Uno dei primi microcontrollori a disporre di un compilatore ANSI C è stato l'8052. Questo microcontrollore, nonostante sia stato uno dei primi disponibili sul mercato (fu introdotto dalla Intel verso la fine degli anni '70), possiede delle caratteristiche che lo rendono ancora oggi molto interessante, nonché particolarmente adatto ad essere programmato in C come vedremo di seguito.

Nel tempo l'8052 è divenuto praticamente uno standard in campo industriale, ed oggi è prodotto da moltissime società in una grande varietà di configurazioni, formati, capacità di memoria e di velocità.

Il compilatore SDCC di cui ci occuperemo in questa puntata è nato proprio per supportare questo microcontrollore, anche se attualmente è in grado di generare codice anche per altri popolari microcontrollori. Esso è quindi uno strumento molto valido ed utile, soprattutto in considerazione del fatto che è disponibile come

freeware, quindi completamente gratuito.
Prima di passare alla descrizione del compilatore è il caso di dare un'occhiata alle caratteristiche dell'8052, per potere comprendere meglio alcune modalità operative ed opzioni del compilatore stesso.

IL MICROCONTROLLORE 8052

Iniziamo col precisare che nonostante di seguito verrà sempre usata la sigla "8052", in realtà quanto detto vale anche per le diverse versioni e derivazioni, che in genere risultano compatibili sia a livello di codice che di funzionalità base. Tra le versioni compatibili includiamo anche il "fratello minore" 8051 (e derivati), che è compatibile a livello di codice, ma ha un numero di periferiche leggermente ridotto.
L'8052 è un microcontrollore ad 8 bit di tipo CISC, dotato di 255 istruzioni, 256 byte di RAM interna, 32 linee di I/O, tre timers, ed una porta di comunicazione seriale. Il programma risiede su un'apposita memoria interna che può essere ampia fino a 64KB, e può essere realizzata con varie tecnologie a seconda del produttore (Flash, EEPROM, EPROM, PROM, RAM...). L'8051 differisce leggermente da queste caratteristiche in quanto possiede solamente due timers e 128 byte di RAM.
Entrambi i microcontrollori sono predisposti per utilizzare fino a 64KB di memoria esterna per memorizzare il codice o i dati (rinunciando ad alcuni piedini di I/O). Alcune versioni, prodotte ad esempio da Philips, possono indirizzare addirittura alcuni MB di memoria! Questa caratteristica amplia moltissimo le possibilità del microcontrollore, anche se bisogna notare che gli accessi alla memoria esterna in genere sono un po' più lenti rispetto a quella interna.

Figura 1: *Mappa di memoria dell'8051/52*

In figura 1 è riportata la mappa di memoria dell'8052. Come si può vedere la memoria interna viene utilizzata per diversi scopi oltre che per memorizzare dati temporanei.
La parte bassa della memoria interna (dall'indirizzo 0x00 a 0x1F) è utilizzata per memorizzare il valore dei registri. Sono disponibili 4 banchi da 8 registri ciascuno (da R0 a R7), ma è possibile selezionarne ed utilizzarne solo uno alla volta. Questa caratteristica risulta particolarmente utile per salvare velocemente il valore dei registri stessi nel caso di interruzioni o di chiamata ad una funzione, o semplicemente per memorizzare un maggior numero di variabili.
L'area di memoria seguente (dall'indirizzo 0x20 a 0x2F) è occupata dalla cosiddetta "memoria dei bit". Quest'area ha la particolarità di essere utilizzata direttamente dalle istruzioni di manipolazione dei bit, e risulta indirizzabile da queste con granularità di 1 bit.
È possibile quindi considerarla come un'insieme di 128 bit consecutivi, ciascuno dei quali può essere indipendentemente posto ad 1 o 0, ed essere utilizzato come flag, o per costruire o decodificare particolari campi di dati relativi a protocolli di comunicazione.
La memoria compresa tra l'indirizzo 0x30 a 0x7F può essere utilizzata dall'utente per memorizzare dati generici e per allocarvi lo

stack. Quest'ultimo è gestito in maniera automatica in caso di chiamata a funzione, ma può essere anche utilizzato manualmente (per passare o memorizzare temporaneamente variabili) grazie alle istruzioni PUSH e POP. Va notato che lo stack cresce nella direzione degli indirizzi crescenti, e la sua posizione iniziale può comunque essere scelta dall'utente.
Gli indirizzi compresi tra 0x80 e 0xFF hanno alcune caratteristiche un po' particolari. Nell'8051, che come già detto è dotato di soli 128 byte di RAM interna, questi ulteriori 128 byte contengono un insieme di registri speciali detti *Special Function Registers* (SFR). Anche nell'8052 questa area è utilizzata dagli SFR, tuttavia gli stessi indirizzi (ma non fisicamente la stessa area di memoria) sono utilizzati anche per indirizzare gli ulteriori 128 byte di RAM.
Quindi apparentemente ad uno stesso indirizzo corrisponde sia un SFR sia una locazione della RAM estesa. L'ambiguità è risolta dall'uso di diverse modalità di indirizzamento: gli SFR si accedono utilizzando la modalità *diretta*, la RAM quella *indiretta*.
Gli SFR sono utilizzati per controllare alcune particolari funzioni del microcontrollore, ad esempio i dati presenti sulle porte di I/O, le impostazioni ed il valore dei timers, e la porta seriale.
Altri SFR sono utilizzati per contenere il valore dell'accumulatore A, del registro ausiliario B, e del puntatore a 16 bit per l'eventuale memoria esterna.
Il numero degli SFR dell'8052 è relativamente piccolo, per cui molte locazioni tra le 128 disponibili non sono utilizzate. Proprio questo fatto è sfruttato dai derivati dell'8052 per aggiungere nuove caratteristiche senza rinunciare alla compatibilità con l'originale: gli SFR dell'8052 originale sono mantenuti, e si trovano alle stesse locazioni, mentre quelli aggiuntivi utilizzano locazioni prima non utilizzate. Questo accorgimento permette in primo luogo di eseguire lo stesso codice su qualsiasi versione del dispositivo (avendo cura di gestire i registri aggiuntivi), ed inoltre rende possibile utilizzare gli stessi tool di sviluppo (assemblatori, compilatori, debugger…) indipendentemente dalla marca, versione, e capacità del microcontrollore utilizzato.

SET DI ISTRUZIONI

Come già detto l'8052 è un microcontrollore CISC (*Complex Instruction Set Computer*), questo significa che oltre a possedere un numero di istruzioni mediamente più alto dei microcontrollori RISC, le singole istruzioni riescono anche ad eseguire funzioni più complesse (cioè a più alto livello). Lo svantaggio di questo approccio è dato dal numero di cicli di clock utilizzati per eseguire ciascuna istruzione: circa 12. Va sottolineato comunque che più recentemente sono stati prodotti degli 8052 in versione RISC o "quasi-RISC", che riescono ad eseguire un istruzione in un numero minore di cicli di clock (da 1 a 4) senza rinunciare alle potenzialità del set di istruzioni originale.
Tra le 255 istruzioni originali le più interessanti e versatili sono quelle di movimento (MOV), che supportano un indirizzamento immediato, diretto e indiretto, sia per la memoria interna che per quella esterna. Tra le istruzioni aritmetiche e logiche invece sono presenti, oltre alle più comuni (somma, sottrazione, incremento…), anche un'istruzione per la moltiplicazione (MUL) ed una per la divisione (DIV), la prima lavora con operandi ad 8 bit e restituisce un risultato a 16 bit, la seconda utilizza due operandi ad 8 bit e restituisce contemporaneamente il risultato della divisione ed il resto in due registri separati.
Tra le istruzioni relative ai salti spicca la famosa istruzione DJNZ (Decrement and Jump if Non Zero), che permette di implementare dei cicli in maniera estremamente efficiente, utilizzando come contatore il valore caricato in un registro. Si rivelano di grande utilità anche le istruzioni per la manipolazione dei bit SETB, CLR, CPL, usati rispettivamente per settare, azzerare o complementare un bit della memoria dei bit o in un SFR.
La ricchezza dei modi d'indirizzamento, la possibilità di utilizzare memoria esterna, di gestire uno stack anche con istruzioni dedicate (PUSH e POP), e la disponibilità di un completo set di istruzioni aritmetiche, fa dell'8052 un target ideale rispetto ad altri microcontrollori per l'esecuzione di programmi C, dal momento che tradizionalmente i compilatori modellano il codice

creato proprio su queste caratteristiche. Questo fa si che il codice generato per l'8052 sia mediamente molto efficiente sia dal punto di vista della velocità che dell'occupazione di memoria.

IL COMPILATORE SDCC

Dopo questa breve introduzione sull'8052, torniamo a parlare del compilatore vero e proprio. L'SDCC (Small Devices C Compiler) è un compilatore ANSI C a linea di comando scritto da Sandeep Dutta, dedicato all'8052, ma che come già detto è utilizzabile anche per altri microcontrollori, tra cui Zilog Z80, Motorola HC08, e (in fase di sviluppo) PIC e AVR. Un primo punto di forza di questo compilatore risiede nel fatto che esso è completamente open-source, ed è distribuito gratuitamente sotto licenza GPL. L'SDCC è disponibile attualmente per i sistemi operativi Windows, Linux, e FreeBSD, ma dal momento che i codici sorgenti sono liberamente accessibili (guarda caso scritti in C!) è facile compilarlo anche per altre macchine o sistemi operativi.
È possibile scaricare la versione più recente (2.4.0) dal sito Internet: http://sdcc.sourceforge.net/snap.php.

Il compilatore è accompagnato da una serie di tool estremamente utili ed interessanti, tra cui l'assembler (per i diversi processori), il linker, un simulatore 8051, ed un debugger funzionante a livello di codice.
SDCC supporta la maggior parte delle caratteristiche dell'ANSI C standard, ed in particolare supporta nativamente i tipi di dati interi **char** (8 bit), **short** e int (16 bit), **long** (32 bit), ed anche i tipo **float** a singola precisione (32 bit) utilizzando la rappresentazione standard IEEE754. È supportato anche l'in-line assembler ed alcune estensioni ed ottimizzazioni dedicate ai diversi microcontrollori. Ad esempio è possibile specifi-

care direttamente nel codice C (con delle keyword dedicate) se una variabile o un puntatore deve essere allocata nella RAM interna, in quella esterna, nella memoria del codice, nella memoria dei bit o nell'area degli SFR. È possibile inoltre specificare che una funzione deve essere considerata la routine di servizio delle interruzioni, o che è una funzione "critica", cioè che non deve essere interrotta da interruzioni.
Le funzioni di libreria standard più utili sono supportate, altre sono in fase di sviluppo, altre ancora non sono supportate completamente (a causa della difficoltà o della poca utilità dell'implementarle su un microcontrollore).
Il compilatore esegue diverse ottimizzazioni abbastanza avanzate per generare il codice finale, tra cui l'ottimizzazione del loop, l'eliminazione del codice "inutile", l'uso di tabelle per le istruzioni *switch*, l'eliminazione delle sottoespressioni comuni, ed alcune semplificazioni algebriche. Inoltre dispone di un ottimizzatore "peep-hole", che esegue la sostituzione di alcuni pattern con altri più efficienti utilizzando una serie di regole.
Una funzione interessante è quella che permette di calcolare la complessità di una funzione (*cyclomatic complexity*). Utilizzando questa opzione è possibile analizzare i propri programmi da questo punto di vista, in modo da mantenere il grado di complessità degli algoritmi il più basso possibile e quindi ottenere funzioni più affidabili.

INSTALLAZIONE E TEST

Come già detto SDCC è disponibile per diversi sistemi. In ambiente Linux/Unix è possibile scaricare i codici sorgenti e compilare il tutto per la propria piattaforma, nel caso di utilizzo in ambiente Windows invece è più semplice scaricare direttamente il pacchetto d'installazione che contiene tutti i programmi già compilati e pronti per essere utilizzati. In entrambi i casi tra i file scaricati è presente una ben curata e molto completa documentazione (in diversi formati, tra cui PDF ed HTML).
Nel caso di utilizzo sotto Windows il programma d'installazione provvederà automaticamente ad includere la directory in cui risiede il compilatore nella variabile d'ambiente PATH (questo significa che per rendere effettive le modifiche è necessario riavviare il sistema, oppure digitare manualmente la riga al prompt dei comandi).
Una volta installato il programma il manuale suggerisce di eseguire una prima prova di compilazione, creando il seguente piccolo file (test.c):

```
char test;

void main(void) {
   test=0;
}
```

ed invocando dalla linea di comando:

sdcc –c test.c

Se non si specificano altri parametri il compilatore genera per default codice per 8052. L'opzione –c serve per evitare di utilizzare il linker, in questa prima fase. Se l'installazione è andata a buon fine il compilatore non dovrebbe dare nessun messaggio di errore o warning. A questo punto è possibile provare a rimuovere l'opzione –c, in modo da testare anche il linker. Se ancora non si ottengono messaggi di errore è possibile provare anche le librerie, utilizzando il seguente codice:

```
#include <string.h>
char str1[10];

void main(void) {
   strcpy(str1, "testing");
}
```

digitando alla linea di comando:

sdcc test.c

se anche questa volta non si ottengono errori, il compilatore è stato correttamente installato ed è pronto per l'uso.

È interessante osservare i file generati dopo il comando di compilazione (figura 2). Tra questi i

Figura 2: *File generati dopo la compilazione di Test.c*

più interessanti sono il file test.asm, che contiene il codice assembler prodotto dal compilatore, il file test.lst, che è il *listing* generato dall'assembler, e contiene affiancati la traduzione in codici macchina (esadecimali) dell'assembler, gli indirizzi in cui verranno memorizzati i codici, e le istruzioni originali. Sono anche riportati, sotto forma di commenti, le linee di codice C a cui l'assembler si riferisce. Questa caratteristica risulta sia utile (nel caso di debug o profiling), sia molto istruttiva, in quanto permette di comprendere ed apprezzare il comportamento del compilatore. Il file test.mem contiene un sommario sull'utilizzo delle risorse di memoria del processore, mente il file test.ihx è il codice oggetto generato dopo l'intero processo, in formato Intel hex (è possibile selezionare altri formati di output). Prima della fase di link, viene generato anche il file test.rel: si tratta del codice oggetto in formato rilocabile, che è utilizzato dal linker per creare il codice oggetto finale.

CODICE DI ESEMPIO

Nel caso di programmi con una struttura più complessa, ad esempio con codice su più file, o che utilizzi librerie esterne, la procedura di compilazione è leggermente diversa. L'SDCC infatti è in grado di processare un solo file sorgente alla volta. Questo implica che i vari file che compongono un progetto devono prima essere compilati separatamente, disabilitando il linker, e poi compilati assieme al file che contiene la funzione *main* (per essere linkati). Ad esempio, se un progetto è composto dai tre file Principale.c, Aux1.c, Aux2.c, e la funzione *main* è contenuta in Principale.c, si procederà come segue:

```
sdcc –c Aux1.c
sdcc –c Aux2.c
sdcc Principale.c Aux1.rel Aux2.rel
```

Quando si hanno molti file secondari che svolgono funzioni di libreria può essere utile, al fine di riutilizzarli in seguito, riunirli in un unico file di libreria propriamente detto. Per fare questo si può usare il programma **sdcclib**. Occorre anche in questo caso prima compilare i vari file sorgenti ed ottenere i file rilocabili .rel, poi creare una libreria (file .lib) aggiungendoli singolarmente. Sarà sufficiente scrivere:

```
sdcc –c Aux1.c
sdcc –c Aux2.c

sdcclib mialib.lib Aux1.rel
sdcclib mialib.lib Aux2.rel
```

Per vedere quali files e quali funzioni contiene la libreria si può digitare:

```
sdcclib –s mialib.lib
```

Per utilizzare la libreria in un programma basterà scrivere:

```
sdcc Principale.c mialib.lib
```

Per eseguire una prima prova "su strada" del compilatore di seguito è riportato un piccolo codice di esempio.
Il programma pilota la porta P1 in modo da fare accendere in sequenza e nelle due direzioni dei LED collegati alle 8 linee (il famoso effetto "Supercar" per intenderci).
Il codice è suddiviso in più file, quello chiamato Led.c è il modulo principale, e contiene il *main*, il file wait.c contiene soltanto una semplice funzione di ritardo, mentre il file Reg52.h è utilizzato

per definire alcune delle locazioni degli SFR dell'8052 (notare l'uso dell'apposita keyword *sfr*).

Contenuto del file Led.c

```
#include "REG52.H"
#include "wait.h"

void main (void)  {
  unsigned char j;

  // *** Ciclo infinito ***
  while (1) {

    /* Scorrimento verso SX */
    for (j=0x01; j != 0x80; j<<=1) {
      /* Accende LED su Porta 1 */
      P1 = j;
      /* Breve attesa */
      wait ();
    }

    /* Scorrimento verso DX */
    for (j=0x80; j != 0x01; j>>=1)  {
      /* Accende LED su Porta 1 */
      P1 = j;
      /* Breve attesa */
      wait ();
    }
  }
}
```

Contenuto del file Wait.c

```
#include "wait.h"

/* Funzione wait semplice */
void wait (void)  {
  unsigned int i;

  // Ciclo vuoto
  for (i = 0; i < N; i++)
  { }  ;
}
```

Contenuto del file Wait.h

```
#define N 10000
```

```
/* Ritardo semplice */
void wait (void);
```

Contenuto del file Reg52.h

```
/* Definizione degli SFR */
sfr P0 = 0x80;
sfr P1 = 0x90;
sfr P2 = 0xA0;
sfr P3 = 0xB0;
```

Per compilare il codice è necessario utilizzare i seguenti comandi:

sdcc –c wait.c
sdcc led.c wait.rel

È interessante notare il modo in cui è stato ottenuto lo scorrimento dell'1 sulla porta di uscita: è stato utilizzato un ciclo *for* con dei parametri un po' particolari.
Dapprima la variabile è inizializzata in modo da presentare un bit 1 nella posizione più (meno nel secondo caso) significativa, la condizione di uscita è data dal raggiungimento dell'1 dall'altro lato del byte, e l'incremento è in realtà uno shift. Questo è un ennesimo esempio dell'estrema versatilità del linguaggio C!

CONCLUSIONI

Vorrei concludere questo articolo sottolineando che nonostante l'SDCC abbia delle grandi potenzialità e possa risultare estremamente utile in molti casi (ad esempio quando occorra saltuariamente scrivere qualche routine in C, o per gli hobbysti), non è ancora dotato della robustezza e delle funzionalità avanzate dei compilatori commerciali.
Ad esempio molte funzioni di libreria standard non sono ancora implementate, ed anche le operazioni in virgola mobile supportate sono solo quelle aritmetiche (niente seno, coseno logaritmo…).

Tuttavia trattandosi di un progetto open-source probabilmente queste lacune saranno colmate a breve, così come sarà presto esteso il supporto ad altri microcontrollori.

www.italelectronics.it

SMARTCARD settima parte

APPLICAZIONI CON LE SIM

di Giuseppe Modugno
gppe.modugno@libero.it

In questo ultimo articolo, continueremo la descrizione delle SIM utilizzate nel sistema GSM, commentando i comandi ed i file più importanti. Successivamente, svilupperemo un apposito firmware per il microcontrollore PIC16F628 del lettore UniReader per la gestione di smartcard asincrone ISO7816. Infine, mediante un software per PC ad hoc, scritto in Visual Basic, potremo interagire direttamente con una SIM senza l'utilizzo di un telefonino, per esempio per salvare sul nostro computer il contenuto della rubrica e gli SMS.

RIPRENDIAMO LA NOSTRA PARTITA A CARTE

Cosa ci siamo detti negli articoli precedenti

Siamo arrivati all'ultima parte di questo lungo tutorial sulle smartcard. Per coloro che non ci hanno potuto leggere sin dall'inizio, facciamo qui un breve riassunto sul contenuto degli articoli precedenti.

Nella prima parte del tutorial, abbiamo introdotto le smartcard da un punto di vista teorico, rispondendo a domande del tipo: cosa sono, quanti tipi ne esistono e come si classificano, quali sono le normative internazionali di riferimento, ecc.

Nella seconda parte è stata descritta in dettaglio una particolare tipologia di smartcard, utilizzata in molte applicazioni comuni: le smartcard a memoria. In particolare, è stato analizzato il diffusissimo chip SLE4442.

Nella terza parte, invece, abbiamo analizzato le smartcard a microprocessore, standard e "meno standard", introducendo termini come ATR, convenzione diretta/inversa, comandi ISO7816-3, protocollo di comunicazione T=0, ecc.

Nella quarta parte, abbiamo finalmente concretizzato le conoscenze acquisite nei precedenti articoli, progettando e realizzando un economico e semplice da costruire lettore universale di smartcard a memoria e a microprocessore, basato sul PIC16F628, chiamato UniReader.

Potete trovare due esempi tipici di applicazione delle smartcard a memoria SLE4442 nella quinta parte del tutorial, in cui è stata mostrata la versatilità di UniReader come lettore slave di un PC o stand-alone.

Infine, nel precedente articolo (sesta parte), abbiamo iniziato la descrizione di una particolare tipologia di smartcard a microprocessore, le SIM del sistema GSM. Abbiamo descritto brevemente le normative ISO7816-4 che le SIM seguono. Inoltre, abbiamo fatto una breve panoramica sui problemi di autenticazione e segretezza delle informazioni nella rete GSM, garantiti proprio dalla SIM.

DESCRIZIONE DEI COMANDI

Ci siamo lasciati con un lungo elenco di comandi previsti dalle normative che tutte le SIM devono correttamente interpretare. I più importanti saranno qui analizzati.

Byte	Descrizione	Lunghezza
1-2	Riservati per usi futuri	2
3-4	Spazio disponibile nella directory non allocato ad alcun EF o DF	2
5-6	File ID	2
7	Tipo di file (01 per l'MF, 02 per DF)	1
8-12	Riservati per usi futuri	5
13	Lunghezza dei dati che seguono	1
14	Caratteristiche del file	1
15	Numero di DF contenuti nel DF selezionato	1
16	Numero di EF contenuti nel DF selezionato	1
17	Numero di codici segreti (CHV e simili)	1
18	Riservato per usi futuri	1
19	Stato CHV1	1
20	Stato UNBLOCK CHV1	1
21	Stato CHV2	1
22	Stato UNBLOCK CHV2	1
23	Riservato per usi futuri	1
24-34	Riservato per scopi di gestione della SIM	0÷11

Tabella 1: *Struttura della risposta al comando SELECT per un DF*

SELECT

È il comando che permette di selezionare un file (EF o DF) della SIM, specificando il suo identificatore a due byte. I file che è possibile selezionare sono quelli che seguono le regole descritte nel precedente articolo. Ricordo che dopo l'ATR la directory selezionata è l'MF. Per poter selezionare, per esempio, il $DF_{TELECOM}$ (7F10) che è contenuto nell'MF, è necessario inviare alla SIM il seguente APDU: A0 A4 00 00 02 7F 10.
Come indicato nella Tabella 3 del precedente articolo, il comando prevede un invio di dati dall'ME alla SIM (l'identificatore del file di due byte da selezionare, 7F10 nell'esempio di $DF_{TELECOM}$) ed una risposta da parte della SIM. Per ottenere questa risposta, di lunghezza variabile, è necessario utilizzare il comando GET RESPONSE. Essa riporta il tipo di file appena selezionato, i suoi attributi (dimensione totale, dimensione record, ecc), le modalità d'accesso ed altre informazioni. Essa ha una diversa struttura a seconda che venga selezionato un DF oppure un EF. Nel primo caso, la sintassi della risposta al comando SELECT è mostrata nella Tabella 1: essa è formata da almeno 22 byte ed al massimo 34. I campi in grigio sono opzionali e potrebbero non essere trasmessi dalla SIM.

Il byte 14 indica, tra l'altro, se la SIM prevede una modalità di funzionamento in cui il clock può essere bloccato per diminuire il consumo di energia (molto importante in apparecchi portatili come un telefonino). Inoltre, se il bit più significativo di questo byte vale 1, il CHV1 (PIN) è disabilitato; al contrario, se tale bit è nullo, il CHV1 è abilitato.
I byte dal 19 al 22 forniscono maggiori informazioni relative ai codici segreti CHV1, UNBLOCK CHV1, CHV2 e UNBLOCK CHV2. Nei quattro bit meno significativi di ogni byte, è memorizzato il

Byte	Descrizione	Lunghezza
1-2	Riservati per usi futuri	2
3-4	Dimensione del file	2
5-6	Identificatore del file	2
7	Tipo di file (04 nel caso di un EF)	1
8	Riservato per usi futuri (il bit più significativo, nel caso di cyclic file, indica la possibilità di usare il comando INCREASE)	1
9-11	Condizioni d'accesso	3
12	Stato del file	1
13	Lunghezza dei dati successivi	1
14	Struttura dell'EF	1
15	Lunghezza di un record (obbligatorio nel caso di linear fixed e cyclic file)	1
16 e seg.	Riservati per usi futuri	-

Tabella 2: *Struttura della risposta al comando SELECT per un EF*

numero di tentativi rimasti per l'inserimento del relativo codice segreto (per i due CHV sono al massimo 3, per i due UNBLOCK CHV sono al massimo 10). Se questi quattro bit sono tutti nulli, il relativo codice segreto è bloccato: se si tratta di un CHV, si potrà usare l'UNBLOCK CHV corrispondente per lo sblocco, altrimenti la SIM dovrà essere restituita al gestore, poiché non più utilizzabile dall'utente.
Nel caso di selezione di un EF, la risposta della SIM al comando SELECT è formata da un minimo di 14 byte (15 byte nel caso di linear fixed e cyclic file), come mostrato in Tabella 2. Come al solito, i campi su sfondo grigio sono opzionali e potrebbero non essere trasmessi dalla SIM.

La dimensione del file (byte 3-4) indica il numero dei byte di un transparent file, oppure il numero dei record moltiplicato per la dimensione di un record nel caso di un linear fixed file e di un cyclic file. Le condizioni d'accesso (byte 9-11) indicano i permessi necessari per effettuare le operazioni sul file. Il byte 9 indica i permessi per il comando UPDATE (nibble meno significativo) e READ/SEEK (nibble più significativo). Il byte 10 contiene i permessi del comando INCREASE (nibble più significativo). Il byte 11 indica i permessi per il comando INVALIDATE (nibble meno significativo) e per il comando REHABILITATE (nibble più significativo). Il nibble che rappresenta il permesso relativo ad un comando può valere "0" per accesso sempre consentito, "1" per accesso consentito solo dopo aver inserito correttamente il CHV1, "2" per accesso consentito solo dopo aver inserito correttamente il CHV2, "F" per accesso mai consentito.
La struttura del file (byte 14) vale "00" per transparent file, "01" per linear fixed file, "03" per cyclic file. Infine, nel caso di file basato su record, il byte 15 indica la lunghezza (in byte) di un singolo record, altrimenti vale "00".

STATUS
La risposta a questo comando è identica alla risposta ottenuta dal comando SELECT sulla directory corrente.

READ BINARY
È il comando che permette di leggere i dati memorizzati all'interno di un transparent file. I parametri dell'intestazione (P1 e P2) indicano l'offset del primo byte da leggere (a partire da 0), mentre il byte LEN indica il numero di byte da leggere. L'APDU di risposta contiene i byte

desiderati dell'EF correntemente selezionato. Tale comando non può essere usato sui linear fixed file e sui cyclic file.
Naturalmente, può essere eseguito solo se il permesso relativo al comando READ è soddisfatto.

UPDATE BINARY
Il comando UPDATE BINARY permette di scrivere una parte dei dati memorizzati in un transparent file.
In questo caso, è il dispositivo di interfaccia che trasmette i dati da memorizzare, mentre la SIM risponde solo con la Status Word.
È necessario soddisfare il relativo permesso di UPDATE.

READ RECORD
Il comando READ RECORD permette di leggere un record dal file corrente di tipo linear fixed o cyclic. Ovviamente, è necessario che sia soddisfatta la condizione d'accesso in lettura relativa al file.
Il record da leggere dipende dai parametri P1 e P2 dell'header

P1	P2	Modalità	Descrizione
00	02	Next Record	Legge/scrive il record successivo a quello corrente.
00	03	Previous Record	Legge/scrive il record precedente a quello corrente.
n	04	Absoulte/Current Record	Legge/scrive il record corrente (n=0) oppure il record numero n (n≠0) a partire dal primo. Il record corrente non viene modificato.

Tabella 3: Modalità di lettura/scrittura di un record mediante i comandi READ/UPDATE RECORD

del comando, così come descritto nella Tabella 3.

Selezionando un file, non c'è alcun record corrente: in questo caso, la modalità Next restituisce il primo, quella Previous l'ultimo record. In un linear fixed file, se è selezionato il primo record ed è usata la modalità Previous oppure se è selezionato l'ultimo record ed è usata la modalità Next, nessun record verrà letto e la SIM risponderà con una Status Word d'errore (**94 02**). Invece, in un cyclic file, verrà letto il primo record con la modalità Next se l'ultimo record è quello corrente; verrà letto l'ultimo record con la modalità Previous se il record corrente è il primo. Il record corrente viene modificato solo con le modalità Next e Previous Record.

UPDATE RECORD

Il comando UPDATE RECORD modifica il contenuto di un record all'interno del file corrente (linear o cyclic), sempre che siano verificati i permessi di scrittura. Anche per questo comando è possibile usare le modalità descritte in Tabella 3 relative al comando READ RECORD. A differenza di questo, per i cyclic file è possibile usare solo la modalità Previous. In questo caso, il record meno recente sarà modificato e diventerà il record numero 1 della lista.

SEEK

Il comando SEEK permette di ricercare un record all'interno di un linear fixed file che inizia con una particolare sequenza di byte. E' necessario avere i permessi di lettura per poter utilizzare questo comando.

Esistono due tipi di ricerca: il primo tipo modifica solo il record corrente, ma non prevede alcuna risposta da parte della SIM; il secondo tipo, oltre a modificare il record corrente, trasmette in risposta il numero di record appena trovato. Se la ricerca non ha successo, il record corrente non viene modificato e la SIM risponderà con la Status Word d'errore (**94 04**). La massima lunghezza della stringa da ricercare è 16 byte ma, naturalmente, non può superare la dimensione di un intero record. Esistono quattro modalità di ricerca, elencate nella Tabella 4.

Il nibble più significativo di P2, indicato in Tabella con "x", vale 0 per il primo tipo, 1 per il secondo tipo del comando SEEK. Come al solito, se non c'è alcun record corrente, la ricerca parte dal primo record se P2=x2, dall'ultimo record se P2=x3. È il comando che viene usato dal telefonino quando si accede alla rubrica della SIM per ricercare un nome che inizia con una determinata sequenza di lettere.

VERIFY CHV

Tutti noi sappiamo che, all'accensione del telefonino, dobbiamo effettuare una semplice, ma molto spesso noiosa, operazione che è quella dell'inserimento del PIN (sempre che non sia stato precedentemente disabilitato). Quello che realmente succede è l'invio, da parte dell'ME, del comando VERIFY CHV alla SIM.

Il comando prevede l'invio degli 8 byte che formano il CHV, mentre il byte P2 indica il codice da verificare: CHV1 se P2=01, CHV2 se P2=02. Se il codice trasmesso è uguale a quello memorizzato nella SIM, il numero di tentativi di inserimento viene inizializzato al valore 3, altrimenti viene decrementato di una unità. Dopo tre tentativi consecutivi errati, il codice è bloccato ed è necessario utilizzare il comando UNBLOCK CHV insieme al codice di sblocco relativo. Naturalmente il comando viene processato dalla SIM solo se il CHV indicato dal parametro P2 non è disabilitato, né bloccato.

Il CHV è formato da un minimo di 4 ad un massimo di 8 cifre decimali. Ad ogni cifra corrisponde un byte che rappresenta, in

P1	P2	Descrizione
00	x0	Effettua la ricerca a partire dal primo record.
00	x1	Effettua la ricerca a partire dall'ultimo record.
00	x2	Effettua la ricerca a partire dal record successivo a quello corrente.
00	x3	Effettua la ricerca a partire dal record precedente a quello corrente.

Tabella 4: *Modalità di ricerca del comando SEEK*

decimale, quella cifra. I byte rimanenti, nel caso il CHV avesse una lunghezza inferiore ad 8, devono essere posti al valore esadecimale FF.

RUN GSM ALGORITHM

È il comando meno noto all'utente ma più interessante tecnicamente, poiché è quello che permette di effettuare l'autenticazione alla rete GSM durante la fase di inizializzazione, per esempio subito dopo l'accensione. L'ME invia alla SIM, tramite questo comando, il numero casuale RAND di 16 byte ricevuto dalla rete e riceve dalla SIM il numero SRES (4 byte) e la chiave crittografica Kc (8 byte). Il primo numero viene ottenuto eseguendo l'algoritmo A3, il secondo tramite l'algoritmo A8. Il primo permette l'autenticazione della SIM alla rete, il secondo la segretezza delle informazioni scambiate. Il comando viene eseguito dalla SIM solo se è stato verificato il CHV1 e la directory corrente è la DF_{GSM} o una sua sotto-directory.

Potete trovare maggiori informazioni su questo argomento nell'articolo precedente.

Struttura di alcuni file di una SIM

In una SIM GSM sono presenti alcuni file (EF e DF) obbligatori, oltre ad altri file opzionali che possono essere creati liberamente dal gestore della rete. In figura 1 è rappresentata solo una minima parte della struttura gerarchica presente nello standard ETSI, contenente i file che andremo ad analizzare nel corso di questo articolo.

Tralasciamo le cartelle MF ("3F00"), DF_{GSM} ("7F20") e $DF_{TELECOM}$ ("7F10") e analizziamo i tre EF.

EF_{ICCID} (2FE2)

E' un transparent file di 10 byte, contenuto direttamente nell'MF, in cui è memorizzato un numero identificativo della carta, normalmente stampigliato su uno dei lati della SIM. È formato da un massimo di 20 cifre decimali codificate in BCD in 10 byte: i nibble non utilizzati sono riempiti con la cifra esadecimale "F".

È possibile sempre leggere il contenuto di tale file, ma non è possibile scriverlo in alcun caso.

EF_{IMSI} (6F07)

L'EF_{IMSI} è un transparent file, figlio della directory DFGSM, che contiene il codice IMSI della SIM. Il file ha una lunghezza di 9 byte, di cui il primo indica la lunghezza del codice IMSI, in byte, che può essere al massimo di 15 cifre decimali. Come al solito, l'IMSI è codificato in BCD dove i nibble non usati (nel caso di lunghezza inferiore a 15 cifre) sono posti a "F". Il nibble meno significativo del primo byte rappresenta un codice di parità. Naturalmente, per leggere l'IMSI è necessario aver verificato il CHV1, mentre può modificarlo solo il gestore.

EF_{ADN} (6F3A)

È il file che contiene la rubrica dei numeri di telefono memorizzati

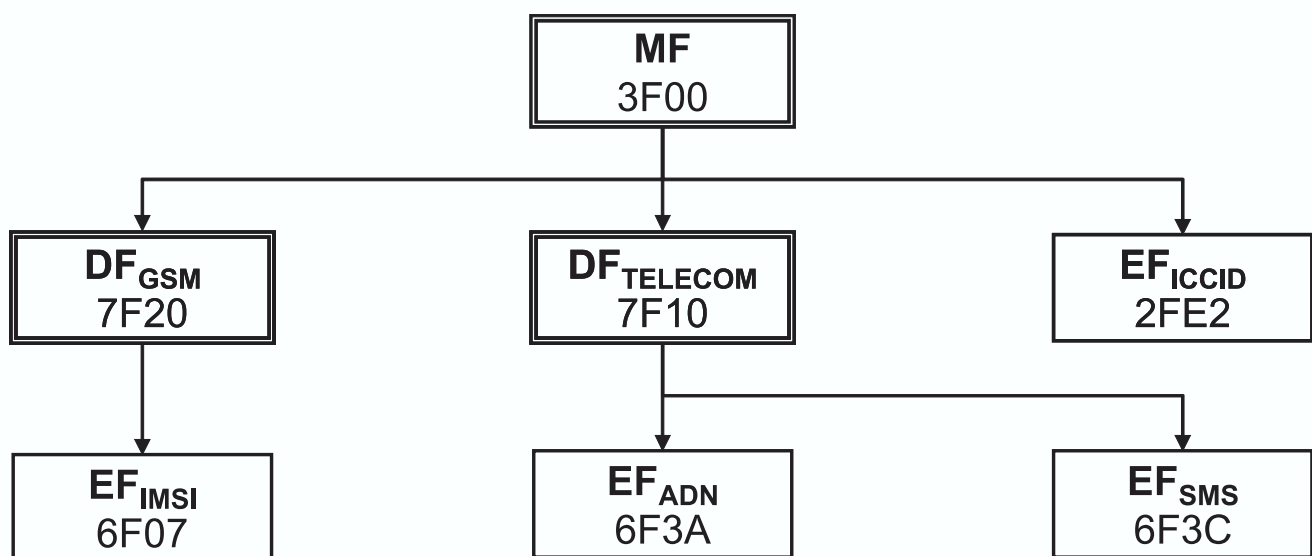
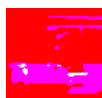



Figura 1: Struttra gerarchica dei file descritti nell'articolo

nella scheda. Il file è strutturato a record in modo sequenziale (linear fixed file) dove ogni record rappresenta un numero di telefono. Naturalmente, per poter leggere e scrivere su questo file è necessario aver verificato il CHV1.
Ad ogni numero di telefono è possibile associare un testo opzionale, normalmente il nome della persona. Se la lunghezza del numero di telefono è fissa (10 byte), lo spazio dedicato al testo associato varia a seconda della SIM.
In Tabella 5 è presente la struttura di un record di questo EF.

Come si può osservare, la lunghezza del record è pari a X+14, dove X può variare a seconda della SIM e rappresenta la lunghezza del testo associato al numero. È possibile conoscere il valore di X sottraendo 14 alla lunghezza di un singolo record, ottenuta dalla risposta al comando SELECT.
I primi X byte contengono l'*alpha-tagging*, cioè la stringa di testo associata al numero telefonico. Faccio notare che il record inizia proprio con il testo e ciò permette di utilizzare il comando SEEK per ricercare un nome in rubrica che inizia con una particolare sequenza di lettere (è praticamente quello che fa il cellulare quando consultiamo la rubrica). La codifica è molto simile all'ASCII, almeno per quello che riguarda i caratteri più diffusi. Rimando il lettore alla lettura dello standard per maggiori approfondimenti.
Il byte X+1 indica la lunghezza, in byte, del numero di telefono, compreso il byte TON+NPI. Tale lunghezza può essere al massimo di 11 byte.
Il byte TON+NPI rappresenta il *Type Of Number* e il *Numbering Plan Identification*. Qui la normativa è piuttosto complessa e non serve scendere troppo nei dettagli. Questo byte è formato, leggendo da sinistra, da un bit pari sempre ad 1, dal TON di 3 bit e dall'NPI di 4 bit. Normalmente il TON è pari a 000 e l'NPI a 0001, così da ottenere il byte **1000 0001**, cioè **81** in esadecimale.
Il numero di telefono è codificato in BCD dove la cifra precedente è codificata nel nibble meno significativo e la cifra successiva nel nibble più significativo. Poiché ogni byte ha due nibble, è possibile memorizzare in un record dell'EF_{ADN} un numero di al massimo 20 cifre. I nibble eventualmente non usati sono messi ad "F". Il simbolo "*" e "#" sono memorizzati rispettivamente con la cifra esadecimale "A" e "B".
Il byte CCI (*Capability/Configuration Identifier*) viene utilizzato se la chiamata al numero telefonico necessita di parametri o requisiti particolari. In questo caso, il byte rappresenta un identificatore del record nel file EF_{CCP} dove sono memorizzate queste informazioni.
Normalmente questo byte non viene usato e, in questo caso, viene messo ad "FF".
Infine, il byte Estensione è utilizzato se il numero telefonico è più lungo di 20 cifre e, quindi, ha una estensione memorizzata in un altro file, denominato EF_{EXT1}. Normalmente non viene usato e viene messo ad "FF".
Per chiarire meglio il tutto, vi sottopongo il seguente esempio di record di un EF_{ADN} dove X=12:

53 65 67 72 65 74 65 72 69 61 FF FF 06 81 83 00 00 44 11 FF FF FF FF FF FF FF

I più attenti dovrebbe risalire al nome ed al numero di telefono associato a questo record (se avete ancora dei dubbi, potete scrivermi).

EF_{SMS}

L'EF_{SMS} è un linear fixed file che contiene i brevi messaggi di testo (SMS) ricevuti e memorizzati nella SIM. Anche in questo caso,

Byte	Descrizione	Lunghezza
da 1 a X	Testo	X
X+1	Lunghezza del numero di telefono (compreso il byte TON+NPI)	1
X+2	TON+NPI	1
da X+3 a X+12	Numero di telefono	10
X+13	CCI	1
X+14	Estensione	1

Tabella 5: *Struttura di un record dell'EF_{ADN}*

la lettura e la scrittura è permessa solo dopo aver verificato il CHV1.
Il record di questo file ha una lunghezza di 176 byte, strutturato come in Tabella 6.

Il primo byte rappresenta lo stato del record. Se il bit meno significativo è nullo, quel record non è utilizzato. Se i tre bit meno significativi valgono 001, il record contiene un SMS ricevuto e letto; se valgono 011, si tratta di un SMS ricevuto ma ancora da leggere. È possibile memorizzare nella SIM anche SMS inviati alla rete ed in questo caso il byte di stato indicherà se l'SMS è già stato inviato, se è stata ottenuta la conferma di invio, ecc. Nel seguito ci soffermeremo solo sugli SMS ricevuti.
Dal secondo byte viene memorizzato il numero del Centro Servizi, utilizzato dalla rete GSM per lo smistamento degli SMS. Come per il numero telefonico memorizzato in un record dell'EF_{ADN}, questo campo è formato da un byte che indica la lunghezza del numero (compreso il byte TON+NPI), un byte che indica il TON+NPI ed infine il numero vero e proprio in formato BCD. Questo campo può avere dimensione variabile, indicata in Tabella 6 con X.
I byte rimanenti contengono il cosiddetto TPDU (*Transfer Protocol Data Unit*), cioè l'SMS opportunamente codificato per

Byte	Descrizione	Lunghezza
1	Stato	1
da 2 a X+1	Numero Centro Servizi	X
da X+2 a 176	TPDU	175-X

Tabella 6: *Struttura di un record dell'EF_{SMS}*

Byte	Descrizione	Lunghezza
1	Primo byte	1
2	Lunghezza in cifre del numero del mittente	1
da 3 a Y+2	Numero del mittente	Y
Y+3	Identificatore del protocollo	1
Y+4	Schema di codifica dei dati	1
da Y+5 a Y+11	Orario e data di invio	7
Y+12	Lunghezza dei dati utente (in caratteri)	1
da Y+13 a 175	Dati utente	-

Tabella 7: *Struttura di un TPDU SMS-DELIVER*

essere trasmesso o ricevuto dalla rete GSM. La struttura del TPDU non è per niente semplice, in quanto lo standard prevede numerosissimi parametri e possibilità, aumentate nel tempo con l'aggiunta di SMS concatenati, immagini, suonerie, ecc. In Tabella 7 è mostrata la struttura del più tipico TPDU denominato SMS-DELIVER, associato ad un SMS in entrata correttamente ricevuto.

Anche in questo caso è inutile scendere troppo nei dettagli. Basti sapere che nel TPDU si trovano codificati il numero del mittente, la data e l'ora in cui il messaggio è stato inviato e preso in consegna dal Centro Servizi, nonché il messaggio vero e proprio (generalmente una stringa di testo). Il tutto si complica poiché lo standard prevede diverse codifiche dei dati (testo) ed anche una forma di compressione: la codifica usata nell'SMS è indicata nel byte Y+4. Nella maggior parte dei casi, tale byte vale "00" ed indica la codifica standard GSM. Un'altra complicazione è dovuta al fatto che il testo, memorizzato nell'ultimo campo, non è codificato come l'alphatagging di un record dell'EF_{ADN} visto in precedenza (fondamentalmente ASCII), ma è utilizzato un apposito algoritmo per trasformare la sequenza di caratteri (7 bit) in byte (8 bit). Chi volesse approfondire questo aspetto, può leggere il sorgente del programma in Visual Basic che verrà illustrato successivamente in questo articolo, oppure leggere direttamente i non semplici standard ETSI per gli SMS (riferimenti in bibliografia).

ESPERIMENTI

SVILUPPO DEL FIRMWARE PER UNIREADER

Per verificare le potenzialità del nostro lettore con le smartcard a microprocessore, ho scritto un firmware che gestisce gli APDU inviati e ricevuti dalla carta asincrona. I comandi provengono, mediante collegamento seriale, da un computer. Il firmware è molto generico e può essere usato con una qualsiasi smartcard asincrona standard che utilizzi il protocollo di trasporto "T=0".
Come al solito, ho utilizzato il compilatore CC5X della B. Knudsen per la stesura del firmware in linguaggio C. Sul sito di Fare Elettronica, potete trovare sia il sorgente, da usare per eventuali modifiche o solo per scopi didattici, sia il file eseguibile per la programmazione del microcontrollore, all'interno del file compresso.
Dopo la prima fase di inizializzazione in cui vengono impostate le direzioni dei pin, la modalità di funzionamento dell'UniReader per la gestione delle carte asincrone e la porta USART, si entra nel ciclo principale in cui il microcontrollore controlla alternativamente e continuamente la presenza di un byte nel buffer di ricezione della porta seriale ed un eventuale bit di start sul pin di I/O ISO7816.
Ogni byte ricevuto dalla smartcard sarà automaticamente trasmesso, in binario, sulla seriale verso il PC. Per questo motivo non è possibile utilizzare un emulatore terminale, poiché i byte in binario verrebbero visualizzati come simboli non facilmente interpretabili. Sarà necessario sviluppare un software apposito per la comunicazione.
I comandi provenienti da PC tramite seriale sono solo di tre tipi e terminano con il carattere di ritorno carrello "↵":

- "r↵", che effettua il reset della smartcard ponendo la linea RST bassa;
- "R↵", che effettua il reset della smartcard ponendo la linea RST alta;
- "xx↵", dove xx rappresenta un byte in esadecimale che verrà trasmesso alla smartcard.

Come al solito, all'accensione di UniReader sarà necessario inviare un segnale di reset per inizializzare la smartcard. Questa risponderà con l'ATR in corrispondenza del corretto livello di reset. Successivamente sarà possibile inviare i byte trasmettendoli in esadecimale ed intervallandoli dal carattere "↵".

SVILUPPO DEL SOFTWARE

Ho sviluppato un software in Visual Basic, denominato SIMReader, che permette di automatizzare e semplificare tutte le operazioni che devono essere effettuate per la lettura dei dati di una carta SIM. Come al solito, potete trovare sul sito di Fare Elettronica sia il pacchetto con i sorgenti, sia il programma di installazione.
In figura 2 è mostrata la schermata iniziale di SIMReader mediante la quale è possibile leggere il numero seriale della smartcard, normalmente stampigliato direttamente su di essa, e il

numero IMSI utilizzato durante la procedura di autenticazione alla rete GSM (vedi articolo precedente).

Prima di tutto, però, è necessario impostare correttamente la porta seriale a cui è collegato l'UniReader (ovviamente il PIC16F628 del lettore deve essere programmato con il firmware UniReader_async.hex). Per fare ciò, basta cliccare sulla scheda delle "Impostazioni" e scegliere (o digitare) la porta seriale utilizzata.
Inserite una SIM all'interno dell'UniReader, magari utilizzando l'apposito adattatore, e accendete il lettore.
Ora è possibile premere i tasti "Leggi" in corrispondenza del numero seriale e del codice IMSI per ottenere questi codici dalla carta.
Nella schermata denominata Rubrica (figura 3) è possibile leggere l'intero contenuto della rubrica memorizzata nella SIM.

Figura 2: *Schermata iniziale di SIMReader*

Figura 3: *Schermata di SIMReader per la lettura della rubrica*

Per fare questo è sufficiente cliccare sul tasto "Leggi". Se la comunicazione avviene correttamente, il software invierà alla SIM, tramite il lettore, gli appositi comandi per la selezione del file EF_{ADN} e per la lettura di tutti i record memorizzati. Potete salvare la rubrica così ottenuta cliccando sul tasto Salva: verrà creato, nella cartella di installazione del programma, il file di testo "rubrica.txt" che potrete consultare con un qualsiasi editor di testo (per esempio, Blocco Note).

La schermata SMS (figura 4) può essere usata per la lettura degli SMS conservati nella SIM. Anche in questo caso, premendo il tasto "Leggi", il software invierà tutti i comandi appositi per la selezione del file EF_{SMS} e la lettura di tutti i record, estraendo per ognuno i campi più importanti: data/ora, mittente, centro servizi, testo. Sottolineo che solo gli SMS del tipo SMS-DELIVER sono correttamente gestiti dal programma, mentre gli altri non saranno visualizzati. Per salvare gli SMS è possibile cliccare sul tasto "Salva" che creerà il file "sms.txt" sempre all'interno della cartella di installazione di SIMReader.

Figura 4: *Schermata di SIMReader per la lettura degli SMS*

Figura 5: *Schermata di SIMReader per l'invio dei singoli comandi*

La schermata che, però, reputo più importante di tutte è quella denominata genericamente Comandi (figura 5). Tramite questa è possibile resettare ed inviare un qualsiasi comando alla SIM o ad una generica smartcard ISO7816 asincrona. In questo modo potrete sperimentare di persona l'invio dei vari comandi alla SIM e ricavare ulteriori informazioni utili che, per motivi di spazio, non sono state commentate in questo articolo.

Per evitare di bloccare la SIM, ho preferito non includere i comandi per l'inserimento del PIN. Per questo motivo, è necessario disabilitare il PIN mediante telefonino prima di utilizzarla con UniReader e SIMReader. In caso contrario, la SIM risponderà con un errore ad ogni tentativo di lettura protetto dal codice segreto.
Naturalmente, se volete, potete completare il software aggiungendo i comandi necessari per l'inserimento del PIN, dopo averlo richiesto all'utente.
Dopo aver utilizzato il programma SIMReader per comunicare con una SIM, potete consultare il file "log.txt", nella cartella di installazione di SIMReader, in cui sono elencati tutti i comandi e le risposte inviate e ricevute dall'UniReader. È molto interessante per poter capire le informazioni scambiate durante le opera-

zioni di lettura IMSI, seriale, rubrica e SMS.

SVILUPPI FUTURI

Come al solito, anche in questo articolo non ho voluto presentare un sistema "chiavi in mano" da riutilizzare senza alcuna modifica. Piuttosto ho cercato di spiegare il funzionamento del sistema GSM relativamente alle SIM in modo che possiate personalizzare in ogni punto sia il firmware per UniReader, sia il software per PC.

Le possibilità per incrementare le funzionalità del sistema sono veramente tante. Mi accontento di elencarvene qualcuna.

Potete inserire nel software SIMReader la possibilità di specificare un PIN in modo che venga automaticamente inviato alla SIM senza dover necessariamente disabilitare tale protezione con un telefonino. Sarà necessario inviare correttamente il comando VERIFY CHV ed interpretarne la risposta. E' un'operazione abbastanza delicata, poiché, se fatta in modo errato, potrebbe bloccare la SIM dopo tre tentativi. Verificate prima di possedere il relativo UNBLOCK CHV e che sia corretto e funzionante.

Un'altra caratteristica che potete aggiungere alla vostra versione di SIMReader è quella di poter modificare, oltre che leggere, il contenuto della rubrica. Anziché inviare comandi READ RECORD, sarà necessario utilizzare il comando UPDATE RECORD. È importante, in questo caso, codificare il nome ed il numero di telefono così come descritto precedentemente ed inviare l'intera sequenza di byte insieme al comando. Si può pensare di avere una rubrica su PC ed una rubrica sulla SIM e di sincronizzare i dati mediante SIMReader. I più bravi potrebbero salvare i dati ricavati dalla SIM all'interno della rubrica di Windows o, al contrario, leggerli da questa per inviarli alla SIM.

Salendo in ordine di difficoltà, potete anche modificare il contenuto degli SMS utilizzando sempre i comandi UPDATE RECORD anziché READ RECORD sul file EF_{SMS}. Qualcuno potrebbe simulare un SMS mai ricevuto, con tanto di data e mittente, e leggerlo sul telefonino come un vero messaggio. La complessità è dovuta alla codifica dei vari campi dell'SMS, tra cui il testo, che è abbastanza contorta. Una attenta lettura degli standard ETSI è d'obbligo.

Infine, potete cercare di estendere la decodifica degli SMS non solo per le tipologie SMS-DELIVER. Anche per questo, è necessario leggere attentamente gli standard relativi.

STANCHI DI GIOCARE?

Sono arrivato alla fine di questo lungo tutorial e spero di aver fornito a molti gli strumenti necessari per lo sviluppo di applicazioni basate su una qualsiasi tipologia di smartcard. Piuttosto che offrire una unica soluzione finita e limitata, ho preferito discutere su tutti gli aspetti di questo meraviglioso mondo dell'elettronica in modo che ognuno di voi possa autonomamente progettare e realizzare il sistema che più gli serve.

Grazie ai molti che mi hanno scritto e continuano a scrivermi per i complimenti e le critiche. Non esitate a contattarmi per qualsiasi informazione o solo per darmi un vostro feedback su quanto avete letto.

Un'affettuoso grazie anche a Mariangela per avermi corretto le bozze

Se avete ancora dubbi o curiosità sul mondo delle smartcard, comunicatele: chissà, in futuro ci potrebbe essere una nuova… partita a carte!

BIBLIOGRAFIA E LINK

ETSI TS 100 977: *Specification of the SIM-ME Interface.*
ETSI GSM 02.17: *Subscriber Identity Modules, Functional Characteristics*
ETSI TS 123 038: *Alphabets and language-specific information.*
ETSI TS 100 940: *Mobile radio interface layer 3 specification.*
ETSI TS 124 011: *Point-to-Point (PP) Short Message Service (SMS) Support on Mobile Radio Interface.*
ETSI TS 123 040: *Technical realization of Short Message Service (SMS).*
ISO7816 parti 1, 2, 3, 4.

ERRATA CORRIGE

Nello schema elettrico dell'UniReader, pubblicato sul numero 231 di Settembre 2004, sono presenti due errori. I condensatori C8 e C10, relativi al MAX232, hanno le polarità invertite. Lo schema di montaggio è, invece, corretto. Grazie a Luca da Perugia che mi ha puntualmente fatto notare il problema.

ALLA SCOPERTA DELL'OPTOELETTRONICA

sesta parte

I FOTOACCOPPIATORI

di Andrea Perilli
aperilli@infinito.it

Dopo aver parlato della fotoresistenza, riprendiamo parlando dei Fotoisolatori più comunemente conosciuti col nome di Fotoaccoppiatori. Vi anticipiamo che tratteremo di due componenti particolari: l'optotransistor e l'optotriac. In questa puntata ci occuperemo dell'optotransistor, tratterremo la sua composizione e il suo funzionamento, vi spiegheremo come si collega e dove impiegarlo. A fine articolo presenteremo un progetto pratico e utile: un contatore di telefonate senza risposta.

CHE COS'È UN FOTOISOLATORE

Il fotoisolatore è componente molto semplice ma nello stesso tempo geniale. Nasce dalla necessità di isolare i circuiti pilota a bassa tensione dai circuiti di potenza ad alta tensione. Infatti la peculiarità di questo componente è di poter isolare due apparati fino a decine di migliaia di volt. Questa caratteristica ha fatto si che questo componente venisse impiegato nella maggior parte dei circuiti in cui vi erano in gioco le due diverse tensioni. Per fare un piccolo esempio: le applicazioni

Figura 1: *Struttura di un fotoaccoppiatore*

che vedono maggiormente impiegati i fotoisolatori (detti anche fotoaccoppiatori) sono, per esempio, schede pilota per servo-relé, schede pilota per motori in corrente continua e alternata e così via.

STRUTTURA DI UN FOTOACCOPPIATORE

Dopo questa piccola premessa introduttiva passiamo a vedere come è composto un opto-transistor.
Possiamo scomporre la struttura di un fotoaccoppiatore in quattro parti principali:

- Supporto isolante.
- Fototransistor.
- Led trasmittente.
- Terminali esterni.

Come mostra la figura 1, sulla base del supporto isolante che ha la forma di un DIL (Dual In Line) a sei Pin, sono poggiati i terminali che fanno capo da un lato, al fototransistor e dall'altro al led. Ai terminali di sinistra è collegato il led, che non è un led comune, come molti potrebbero supporre, bensì un led emittente a raggi infrarossi. Perpendicolarmente al led troviamo un fototransistor che fa capo ai terminali di destra. Ogni qualvolta il led si accenderà, il fototransistor andrà in conduzione.

COME COLLEGARE UN OPTOTRANSISTOR

Dalla figura 2 possiamo notare come si collega un optotransistor per pilotare un carico.
Nell'esempio A quando il led sarà polarizzato, il fototransistor condurrà e accenderà la lampadina.
Nell'esempio B il led è spento quindi il carico è inattivo. In pratica pilotando il led potremo attivare o disattivare il carico collegato a valle del fototransistor.
Naturalmente gli schemi sono solo ideali ma in pratica bisogna innanzitutto calcolare il valore della resistenza per limitare la corrente che attraverserà il led a 15-20 mA. Vale la formula appresa nella prima puntata.
Rinfreschiamoci comunque la memoria:

Rc = (Val-Vled) : 0,015

dove

- **Rc** è il valore espresso in ohm della resistenza di caduta da collegare in serie
- **Val** è il valore della tensione di alimentazione
- **Vled** è il valore dei volt caduta del led da applicare
- **0,015** è il valore espresso in ampere della corrente che attraverserà il led

Bisognerà comunque ricordare che alimentando il fotodiodo con tensioni alternate o con segnali di bassa frequenza dovremo inserire un diodo (vedi D1 nel progetto pratico) per proteggere il led dalla tensione inversa che se supera la soglia dichiarata nelle caratteristiche proprie del componente, può bruciare irrimediabilmente il led.
Riguardo al fototransistor vogliamo solo precisare che è preferibile non collegare direttamente un carico di potenza ai suoi capi ma con esso pilotare un transistor o un darlington che sopporterà meglio l'assorbimento in corrente del carico stesso.
Conclusa la breve parte teorica passiamo alla parte pratica che sicuramente chiarirà le idee su questo componente.

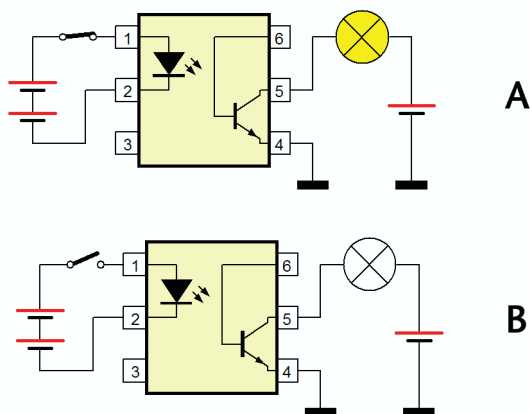



Figura 2: Esempio applicativo di un optotransistor

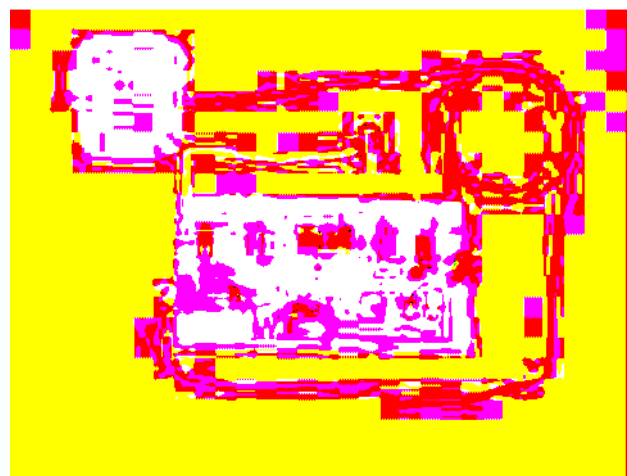



Figura 3: Come si presenta la scheda a montaggio ultimato

LA PRATICA: CONTATORE PER CHIAMATE PERSE

Con questo circuito potrete conoscere il numero delle telefonate giunte al vostro numero durante la vostra assenza.

Il funzionamento è molto semplice: il circuito si collega in parallelo alla linea telefonica ANALOGICA e visualizza, grazie a due display il numero delle chiamate che arrivano.

Il circuito è dotato di alcune funzioni interessanti:

1. Risparmio batteria, infatti in funzionamento normale i due display sono spenti.
2. Accensione temporanea e lampeggiante durante una chiamata.
3. Led lampeggiante che testimonia se vi sono chiamate in memoria.
4. Pulsante di visualizzazione temporanea o istantanea (con C5 inserito o meno).
5. Pulsante di reset per azzerare le chiamate già rilevate.

Lo schema a blocchi di figura 4 ne illustra il funzionamento.

Alla LINEA TELEFONICA è collegato il blocco RIVELATORE DI CHIAMATA, che fornisce alle sue uscite un livello logico alto per ogni chiamata. Il blocco L1 serve ad indicare, senza accendere i display, se vi siano giunte o meno delle telefonate. Grazie al blocco D4 possiamo far lampeggiare i display ad ogni squillo. Il blocco CONTATORE riceve il segnale di clock ed avanza la sua numerazione per ogni chiamata ricevuta, visualizzandone il numero sul blocco DISPLAY. Il blocco P2 serve a visualizzare in modo temporaneo o istantaneo (vedremo più avanti come) il numero delle chiamate sul blocco DISPLAY. Il blocco P1, infine, serve a resettare il blocco L1 e il blocco CONTATORE in modo da cancellare le chiamate già registrate.

UN PO' DI TELEFONIA

Per capire meglio il funzionamento del circuito dobbiamo fare un accenno su come funziona una rete telefonica.

Quando la cornetta è abbassata sulla linea telefonica circola in teoria una tensione continua di 48 volt che in realtà è di 35 - 50 volt (dipende dalle zone), lo potete constatare inserendo il tester sulla portata continua nella presa telefonica. Quando invece alziamo la cornetta e impegniamo la linea la tensione scende a 7 - 8 volt, i quali alimentano i circuiti di fonia. Nel momento in cui una chiamata raggiunge la vostra utenza, sulla linea compare una tensione alternata di 70 - 80 volt a 25 Hz per un tempo di un secondo e mezzo e intervallata da quattro secondi di pausa. Questo segnale serviva all'inizio a far trillare la suoneria meccanica. Al nostro circuito interessa proprio questo segnale di alternata perchè rappresenta una chiamata al nostro apparecchio.

Quando effettuate una chiamata la linea viene chiusa e riaperta con la frequenza del tasto premuto (vi rammentiamo che stiamo parlando di una linea telefonica decadica perché la multifrequenza o DTMF funziona in modo diverso). Per intenderci fate memoria ai vecchi telefoni a disco, quando facevamo girare il disco a comporre lo zero, al suo rilascio la linea telefonica veniva chiusa e riaperta dieci volte in un secondo, quindi a 10 Hz. Per la selezione del nove invece, la linea veniva chiusa e riaperta nove volte in un secondo e cosi via per tutti i numeri. A chi possiede ancora questo modello di telefono proponiamo un semplice esperimento: alzate la cornetta e sui tastini bianchi provate a comporre il numero come si farebbe per un tasto morse, vi accorgerete che il telefono effettua comunque una chiamata. Questo procedimento era usato da molti quando ai telefoni a disco era inserito quel lucchetto che impediva la rotazione del disco stesso.

Vi sveliamo, come nostra consuetudine, un'altra chicca: il suono di

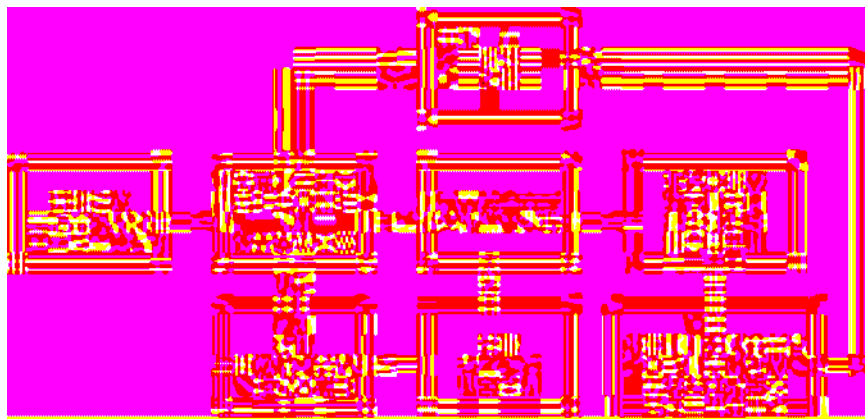



Figura 4: Schema a blocchi del contatore di telefonate

linea libera (il "tu-tu" per intenderci) è la nota LA a 440 Hz.

DECODIFICA E CONTATORE TUTTO IN UNO

Apriamo una piccola parentesi per illustrarvi il funzionamento del 4033, dato che è un dispositivo versatile che potrete usare anche nei vostri progetti.

Il 4033 è un C-Mos contenente un contatore decimale e un driver per display a led a catodo comune, nella figura 5 è riportata la sua piedinatura vista dall'alto, spieghiamo di seguito la funzione dei vari Pin:

Figura 5: *Piedinatura dei display, del 4001 e del 4033*

- **Pin 1 (Ck)**. Questo è il Pin a cui vengono applicati gli impulsi di Clock per incrementare il conteggio. Una cosa importante è da ricordare: il contatore avanza solo con i fronti di salita, ossia quando il clock passa da livello 0 a livello 1 (vedi figura 5).
- **Pin 2 (Ckl)**. Questo Pin è l'abilitazione del segnale di clock. Per far rilevare il clock al contatore, questo Pin dovrà essere collegato a massa.
- **Pin 3 (RBin)**. Questo terminale

serve per escludere o far comparire lo zero, collegandolo a massa non appare mentre se collegato al positivo compare. Di solito nei contatori a più digit questo Pin è pilotato in modo da far apparire o meno lo zero delle decine.

- **Pin 4 (RBout)**. Serve nei contatori a più digit per pilotare l'accensione dello zero.
- **Pin 5 (Cout)**. Questo terminale si usa nei contatori a più digit. Da questo Pin infatti fuoriesce il clock per pilotare il contatore successivo, in pratica quando il contatore delle unità raggiunge il cinque, questo terminale che finora era a livello alto passa a livello basso. All'arrivo della nona unità e al successivo colpo di clock il contatore delle unità si azzera e commuta su questo terminale un livello alto che incrementa il contatore delle decine.
- **Pin 6 (f)**. Cominciano i terminali per il pilotaggio dei segmenti, questo pilota il segmento f.
- **Pin 7 (g)**. Segmento g.
- **Pin 8 (GND)**. Ground, massa: questo terminale va collegato alla massa di alimentazione.
- **Pin 9 (d)**. Segmento d
- **Pin 10 (a)**. Segmento a.
- **Pin 11 (e)**. Segmento e.
- **Pin 12 (b)**. Segmento b.
- **Pin 13 (c)**. Segmento c.
- **Pin 14 (LT)**. Questo piedino va collegato a massa, perché se collegato al positivo provvede all'accensione di prova di tutti i segmenti del display per controllare che non vi siano led bruciati. La sigla LT significa Lamp Test.
- **Pin 15 (R)**. Reset, questo Pin durante il funzionamento normale dovrà essere collegato a massa o forzato a massa. Se applichiamo un livello logico alto a questo terminale il contatore e quindi il display sarà azzerato.
- **Pin 16 (Vcc)**. Questo è il terminale di alimentazione positiva.

Detto ciò possiamo analizzare con più chiarezza come sono stati collegati tra loro i 4033.

SCHEMA ELETTRICO

Cominciamo la descrizione dello schema elettrico di figura 6 da U1, un optotransistor: come abbiamo appreso dalla teoria, al suo interno si trovano un led ed un fototransistor. All'anodo del led interno è collegata R1 che funge da limitatore di corrente, mentre C1 serve a bloccare la componente continua presente sulla rete che altrimenti polarizzerebbe continuamente il led.

Quando la tensione alternata di una chiamata raggiunge i capi del morsetto, attraverso R1 e C1, polarizza il led e manda in conduzione il transistor interno.

Il fototransistor entrando in saturazione induce un livello logico basso sul Pin 5 di U1 che attraverso R2 polarizza la base di TR1. Otteniamo così sul suo collettore un livello logico alto che fungerà da clock per il nostro contatore, C2 ha la sola funzione di antirimbalzo.

Come già accennato una chiama-

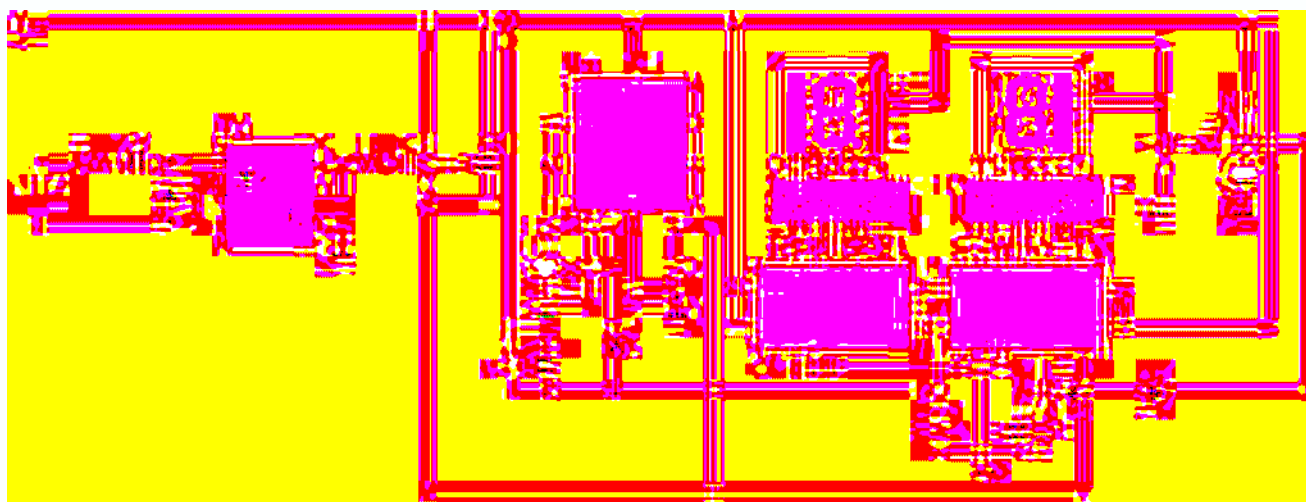



Figura 6: Schema elettrico

ta è composta da un livello di tensione alternata ogni 4 secondi, per evitare quindi che il contatore numerasse gli squilli e non le chiamate abbiamo introdotto un piccolo artificio: R7 e C4 che al primo squillo inibiscono il conteggio, il contatore quindi, per la tutta la durata della chiamata incrementa il conteggio di una sola unità.
Per far lampeggiare L1 quando il contatore avesse memorizzato una chiamata, abbiamo usato U2 un C-Mos contenente quattro porte NOR collegate in configurazione flip-flop Set-Reset.
In pratica il segnale di clock attiva l'ingresso set del flip-flop che commuta la sua uscita sul livello logico alto fino a quando non verrà premuto il pulsante di Reset (P1). Vi sarete accorti che all'uscita del flip-flop si trova la R3 che polarizza L1 facendolo quindi accendere solo se saranno giunte telefonate, in questo modo potrete già sapere a colpo d'occhio se sono giunte chiamate e se quindi c'è bisogno di premere P2 per visualizzarle. Per evitare che all'accensione del circuito i display mostrino dei numeri casuali, abbiamo inserito C3 e R6 che azzerano all'accensione i contatori e il flip-flop.
Siccome l'alimentazione del circuito è erogata da una batteria da 9V (preferibilmente alcalina), per far si che la sua carica duri molto di più, il circuito è stato progettato in modo da lasciare spenti i display durante il funzionamento normale. L'accensione dei display può avvenire manualmente solo quando l'utente vuole controllare il numero delle chiamate registrate. Purtroppo il 4033 non prevedeva la funzione di blanking (l'accensione pilotata del display) quindi bisognava risolvere il problema in maniera indiretta: per accendere manualmente i display basta pilotarne direttamente i loro catodi comuni attraverso TR2, che polarizzato da R8 entrerà in conduzione con la pressione di P2.
Grazie a D4 riusciamo a far accendere i display all'arrivo di una chiamata con la stessa frequenza del ring. Inserendo nelle piazzole previste C5 i display resteranno accesi per tutta la chiamata, in più, successivamente alla pressione di P2 la visualizzazione durerà circa 5 secondi, lasciamo al vostro gusto la scelta delle due diverse opzioni.
Nel caso in cui voleste tenere sempre accesi i display potreste modificare il circuito in due modi:

- Eliminare P2 e cortocircuitare le due piazzole.
- Eliminare P2, D4, R8, C5 (se presente), TR2 e collegare direttamente i catodi dei display a massa, collegando le piazzole di emettitore e collettore di TR2.

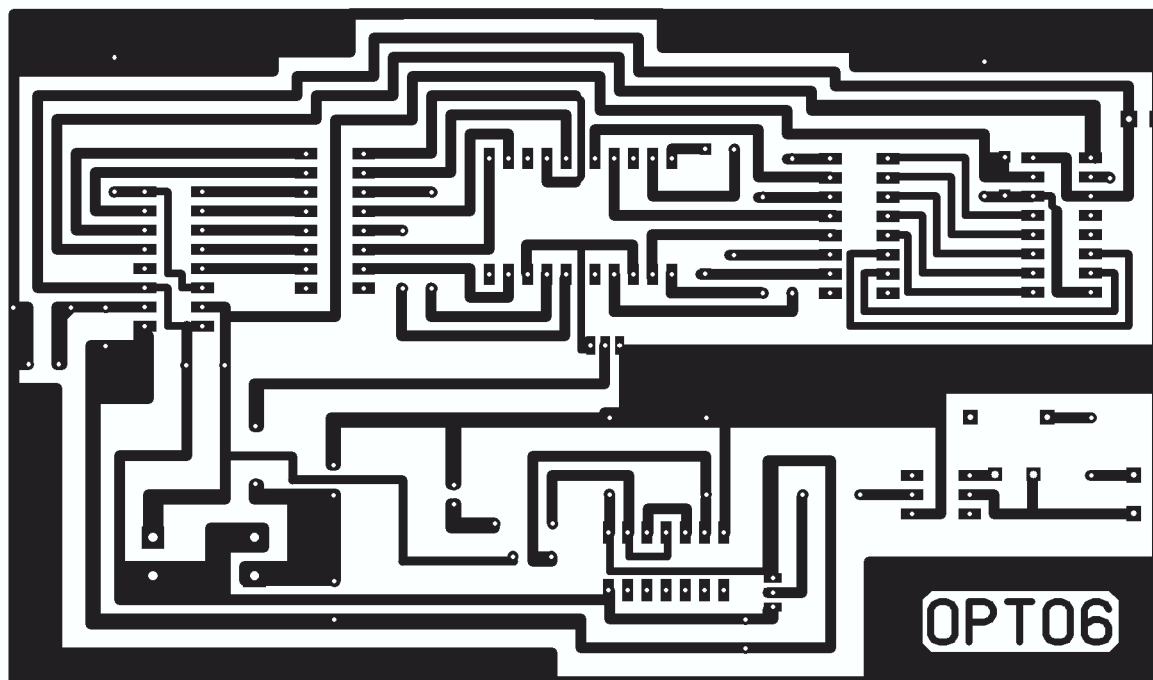



Figura 7: *Lato rame scala 1:1 del circuito*

Figura 8: *Piano di montaggio della scheda*

Per limitare la corrente che scorre nei display usiamo delle reti resistive, che altro non sono che le classiche resistenze però integrate in un package simile ad un circuito integrato, nell'evenienza in cui aveste delle difficoltà a reperirle potete tranquillamente sostituirle con sette resistenze da 470 Ω.
È stato scelto un modello a otto resistenze per eventuali modifiche circuitali da parte vostra, per esempio se vorrete pilotare anche i punti decimali dei display.
Come avete constatato vi diamo

Elenco componenti			
Sigla	**Valore**	**Sigla**	**Valore**
R1÷R3, R8	1 KΩ 1/4 W	U3, U4	CD4033
R4, R5	270 Ω 1/4 W	DS1, DS2	Display Catodo Comune Tipo FND500 o equivalente
R6	100 KΩ 1/4 W	TR1	BC557
R7	15 KΩ 1/4 W	TR2	BC547
RR1, RR2	Rete Resistiva 470 Ω (8 Resistenze singole)	P1	Pulsante N.O. Rosso
C1	1 µF 100 V poliestere	P2	Pulsante N.O. Nero
C2, C3	100 nF 63 V poliestere	N° 1 zoccolo per integrato 6 Pin	
C4	470 µF elettrolitico	N° 1 zoccolo per integrato 14 Pin	
C5 (opzionale)	470 µF elettrolitico	N° 4 zoccolo per integrato 16 Pin	
D1	1N4007	N° 1 morsetto serrafilo 2 vie	
D2÷D4	1N4148	N° 1 Clip per Batteria 9V	
L1	Diodo Led Lampeggiante 5mm Rosso	Doppino telefonico o cavetto bifilare	
U1	Fotoaccoppiatore 4N37 o equivalente	Presa-spina telefonica	
U2	CD4001		

tutte le indicazioni per poter modificare i progetti a vostro piacimento: un altro buon motivo per preferire Fare Elettronica.
Per realizzare un contatore a due digit utilizzando due 4033 bisogna collegare il Pin 5 di U4 con il Pin 1 di U3. Il display in assenza di chiamate visualizza il doppio zero, se questa condizione non vi soddisfa potete scollegare il Pin 3 di U3 dal positivo e connetterlo con la massa, in questo modo il display visualizzerà solo lo zero delle unità.

REALIZZAZIONE PRATICA

Dopo aver realizzato il circuito stampato servendovi del master di figura 7 possiamo procedere al montaggio dei componenti partendo dai nove ponticelli. Vi facciamo notare che abbiamo preferito usare dei ponticelli e non realizzare una basetta doppia faccia poiché la sua realizzazione risulta a molti più difficile e anche perché non tutti gli hobbisti possiedono tutte le attrezzature necessarie già solo per realizzare, ad esempio, i fori passanti metallizzati. Proseguiamo il montaggio con i resistori e i diodi al silicio, notate che D1 è diverso dagli altri, tutti comunque possiedono una fascia colorata che ne indica il catodo.
Successivamente possiamo saldare gli zoccoli per i circuiti integrati facendo riferimento al piano di montaggio di figura 8, in seguito potrete saldare anche gli zoccoli per le reti resistive che non richiedono orientamento.
Passiamo ora a saldare i transistor, attenzione alle sigle perché TR1 è un PNP mentre TR2 è un NPN. A questo punto potrete saldare i display, per questo progetto abbiamo scelto il modello FND500 che nel caso potrete sostituire con un equivalente basandovi sulla piedinatura riportata in figura 5. Fatto ciò possiamo saldare i componenti con profilo più alto: i condensatori in poliestere e gli elettrolitici per i quali dovrete porre attenzione al verso ed infine il led. Per la connessione alla rete telefonica è previsto il morsetto serrafilo a due vie ma volendo potrete saldare direttamente il doppino alla scheda o inserire due capicorda, prima di saldare il clip alla scheda converrà far passare i fili attraverso le asole che impediranno al peso della batteria di staccarne i fili.

COLLEGAMENTI E COLLAUDO

Per collegare il circuito in parallelo alla rete telefonica avrete bisogno di uno spezzone di doppino telefonico e una presa/spina, i cui collegamenti sono illustrati in figura 9. Vi anticipiamo comunque che i terminali da collegare sono quelli con il filo rosso e bianco, la polarità non è obbligatoria poiché al circuito interessa un segnale in alternata. Per il collaudo avremo bisogno di una batteria alcalina a 9V e di un cellulare.
Dopo aver collegato la batteria al clip e la presa/spina nella presa telefonica, potrete chiamare il vostro numero e controllare che il circuito funzioni. È bene comunque specificare che il circuito è stato progettato per numerare solo le chiamate non risposte perché se alzerete la cornetta o lo terrete in funzione rispondendo al telefono la numerazione verrà falsata. Quando la batteria comincerà a scaricarsi potrete rendervene conto da due fenomeni: il primo è che la luminosità dei display e del led è minore e il secondo che il contatore (se per esempio è inserito C5) si azzera con la pressione di P2 oppure conta a due a due le chiamate. Siccome il circuito non possiede una memoria e i numeri in assenza di tensione si azzerano, vi consigliamo di non rimpiazzare la batteria con un alimentatore altrimenti in caso di black-out perderete i dati registrati.

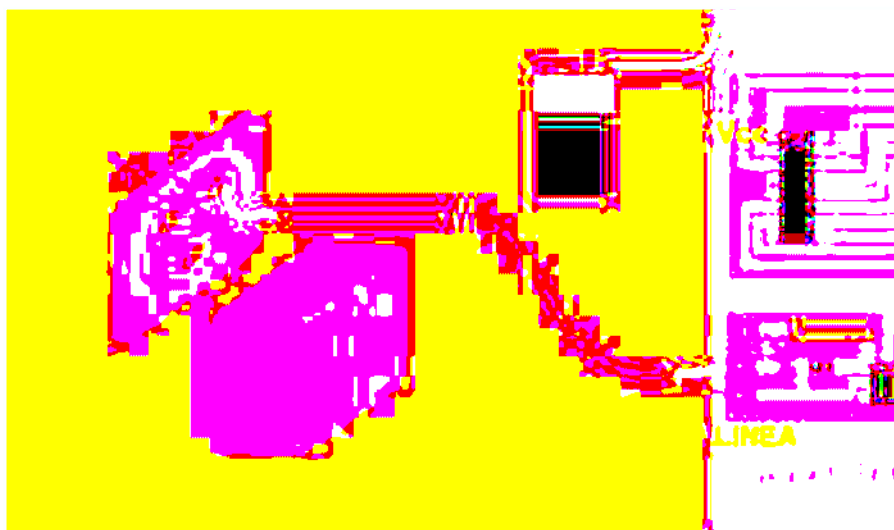
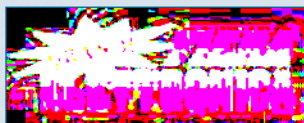



Figura 9: *Collegamenti del circuito alla rete telefonica*

Le fiere e mostre mercato di Dicembre

2004

04-05 Dicembre 2004
GRANDE FIERA DELL'ELETTRONICA

La Grande Fiera dell'Elettronica di Forlì, con due edizioni annuali, è fra le manifestazioni più famose e frequentate a livello nazionale. L'edizione di primavera propone, oltre all'elettronica, il Flight simulator show: gare e dimostrazioni di volo simulato per provare l'ebbrezza di essere alla guida di un veivolo... con i piedi ben a terra! Per gli appassionati di collezionismo radio d'epoca, dischi e CD da collezione, l'appuntamento di dicembre invece è una vera e propria kermesse con oltre 350 aziende partecipanti: elettronica, fotografia tradizionale e digitale, radio d'epoca, dischi e cd da collezione, il concorso nazionale dell'inventore elettrico ed elettronico e il Salone dell'Astronomia che in pochi anni ha conquistato il consenso di espositori e pubblico.

Luogo: Fiera di Forlì – Via Punta di Ferro – Forlì (CO)

Orari: 9.00/18.00

Organizzatore: Blu Nautilus (www.blunautilus.it Tel. 0541-53294)

Ingresso: Intero € 7,00 - ridotto € 6,00

11-12 Dicembre 2004
RADIANTISTICA

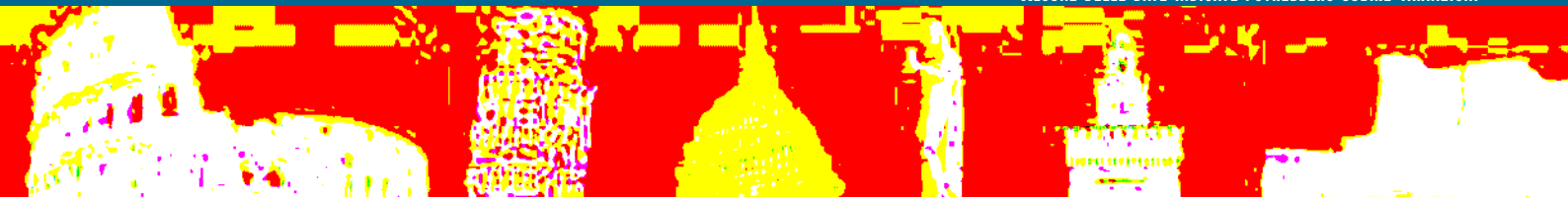
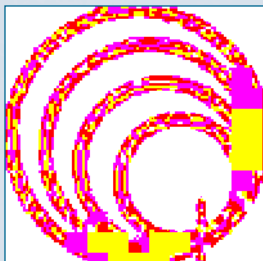
Mostra Mercato Nazionale della Radiantistica Elettronica: materiale radiantistico per C.B. e radioamatori, apparecchiature per telecomunicazioni, surplus, telefonia, computers, antenne e parabole per radioamatori e TV Sat, radio d'epoca, editoria specializzata.
In concomitanza con la fiera ci sarà anche il Salone del Collezionismo (minerali gemmologia, fossili e conchiglie, filatelia e numismatica, oggetti artigianali da collezione) e la Mostra mercato del disco usato in vinile e CD da collezione.

Luogo: Quartiere Fieristico Civitanova Marche (MC)

Orari: 9.00/19.30

Organizzatore: E.R.F. (www.erf.it Tel. 0733-780811)

Ingresso: Intero € 5,00 - ridotto € 3,00

Alcune delle date indicate potrebbero subire variazioni

18-19 Dicembre 2004
24° MARC

Torna in Fiera a Genova il 18 e 19 dicembre il grande appuntamento con MARC, la Mostra di Genova delle attrezzature radioamatoriali organizzata da Studio Fulcro con il patrocinio dell'Associazione Radioamatori Italiani. Un vero e proprio fenomeno di costume che richiama visitatori da tutta Italia alla ricerca di offerte speciali e pezzi rari, da sempre caratteristiche vincenti della rassegna.

Attrezzature e componentistica radioamatoriale sono il nucleo storico di MARC, alle quali si affiancano e crescono informatica, telefonia, sicurezza, accessori e editoria specializzata.

Come di consueto MARC offrirà una panoramica completa di attrezzature e componentistica radioamatoriale, ma anche antenne e parabole satellitari, decoder, stampa specializzata e accessori per i telefoni cellulari. Grandissimo risalto avrà il settore informatico con oltre una trentina di aziende presenti con un'ampia offerta di pc, notebook, stampanti, scanner e software. Mentre i cultori del fai-da-te potranno trovare pezzi di ricambio, apparecchiature da mettere a posto e accessori per elaborazioni speciali a costi estremamente competitivi.

LUOGO: Fiera Internazionale – Genova
ORARI: 9.00/18.30
ORGANIZZATORE: Studio Fulcro (www.studio-fulcro.it - Tel. 010-561111)
INGRESSO: Intero € 7,00 - ridotto € 5,00

22-23 Gennaio 2005
GRANDE FIERA DELL'ELETTRONICA

Ogni anno Exporadio Elettronica alla Fiera di Modena "apre" il calendario delle fiere di elettronica e radiantismo.

L'appuntamento modenese si preannuncia ricco di espositori e prodotti, come sempre tantissimi e di vario impiego: computer, software, periferiche, telefonia fissa e mobile, video games, home entertainment, ricezione satellitare, piccoli elettrodomestici, accessori, ricambi e strumentazioni varie.

Altrettanto assortito il settore del "fai da te" con kit per auto costruzione, surplus, schede, circuiti, e ricetrasmittenti OM, CB, componenti per l'utilizzo radioamatoriale.

LUOGO: Ente Fiera – Modena
ORARI: 9.00/18.00
ORGANIZZATORE: Blu Nautilus (www.blunautilus.it - Tel. 0541-53294)
INGRESSO: Intero € 7,50 - ridotto € 6,50

PRATICAMENTE... I FLIP FLOP: COMANDARE DUE LUCI CON UN SOLO PULSANTE

di Maurizio Del Corso
m.delcorso@farelettronica.com

Il nome è senza dubbio simpatico, ma cosa sono i FLIP-FLOP (FF)? Come funzionano? Quale è la loro utilità? Questo mese risponderemo a queste domande analizzando, come esempio pratico, un dispositivo in grado di gestire l'accensione di due lampade utilizzando un solo pulsante.

ANALISI E SPECIFICHE DEL PROBLEMA

Si deve realizzare un dispositivo capace di gestire l'accensione di due lampade a 12V (L0 e L1) utilizzando un solo pulsante. Inizialmente tutte le lampade dovranno essere spente e, premendo consecutivamente il pulsante, dovranno accendersi con la sequenza seguente: L0 accesa e L1 spenta, L0 spenta e L1 accesa, entrambe accese, entrambe spente. La sequenza si ripete ciclicamente.

LA SOLUZIONE PROPOSTA

La soluzione proposta è riportata nella figura 1 e impiega un integrato 74LS112 che contiene una

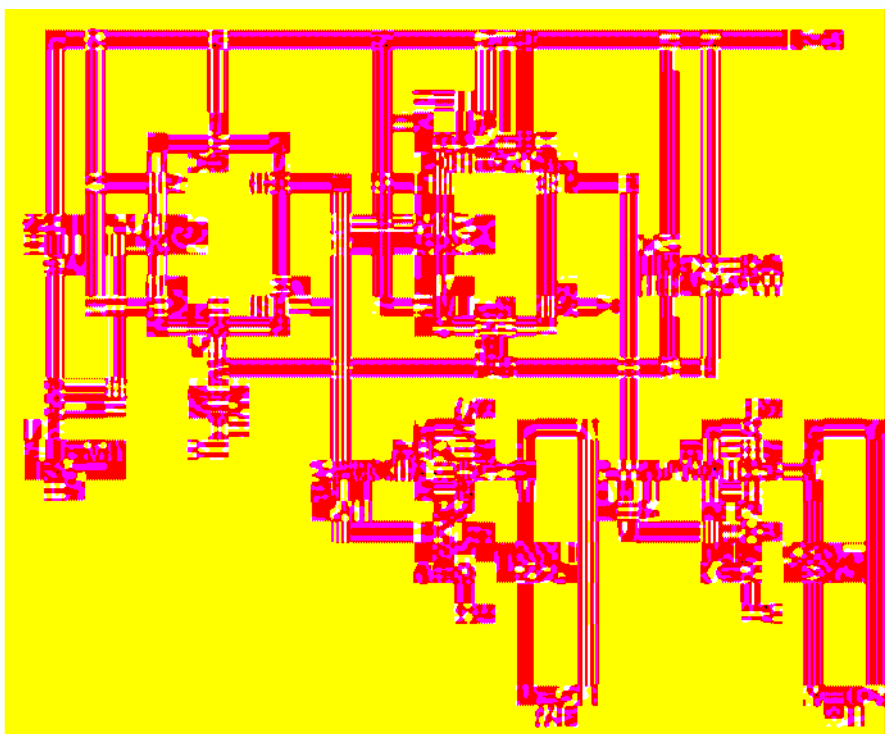



Figura 1: La soluzione proposta

coppia di flip-flop JK.

Le uscite dei due flip-flop comandano dei relé mediante un transistor la cui configurazione è stata già approfondita nella puntata di "Praticamente" pubblicata sul numero 225 di Fare Elettronica (Marzo 2004).
La rete composta da R1 e C1 permette l'azzeramento dei flip-flop all'accensione. Il condensatore C1 è infatti inizialmente scarico e all'accensione mantiene a zero l'ingresso CD dei due flip-flop provocandone il reset. C1 inizia però a caricarsi attraverso R1 per cui, dopo un certo tempo (qualche millisecondo) la tensione ai suoi capi raggiunge un valore tale da essere riconosciuto come "1" logico e i due FF, oramai azzerati, iniziano a funzionare correttamente. Il diodo D1 entra in funzione nella fase di spegnimento e, cortocircuitando R1, provoca la scarica istantanea del condensatore. L'ingresso SD dei FF viene usato per forzare a livello alto le uscite dei FF. Poiché in questa applicazione tale funzione non è necessaria, i due ingressi vengono resi inattivi collegandoli a livello alto (+5V).
Il comando dell'accensione delle lampade avviene mediante il pulsante SW1 il quale, una volta premuto, porta a livello basso l'ingresso CP del primo FF. Se SW1 non è premuto, CP viene mantenuto a livello alto dalla resistenza di pull-up R2. Le resistenze R3 ed R4 limitano la corrente di base dei transistori assicurando che questi vadano in saturazione quando la corrispondente uscita del FF è a livello alto. Per il dimensionamento di tali resistenze è necessario conoscere la massima corrente erogata all'uscita del FF sul livello alto (+5V) che, per il 74LS112 vale 0,4mA. Con il transistor in saturazione si misura una tensione tra base ed emettitore pari a circa 0,7V per cui la corrente che entra nella base del transistor vale:

$$I_B = \frac{5V - 0.7V}{R3} = 0{,}28mA$$

che è sufficientemente al di sotto della massima corrente erogabile dal dispositivo. La saturazione del transistor provoca l'eccitazione del relé il quale chiude il circuito della relativa lampada. VL0 e VL1 sono le tensioni di alimentazione delle lampade e sono determinanti nella scelta del relé. D1 e D3 sono i diodi di libera circolazione che salvaguardano il transistor nella fase di diseccitazione del relé.

COSA È UN FLIP-FLOP

Un flip-flop è una rete digitale sequenziale la cui uscita Q, ad un certo istante di tempo, dipende sia dal valore degli ingressi allo stesso istante, sia dal valore che l'uscita aveva nell'istante di tempo precedente. In figura 2 è riportato un FF di tipo JK e la relativa tabella di verità. Con *n* si è indicato l'istante di tempo, per cui possono verificarsi i seguenti casi:

- J=0, K=0: è detto stato di conservazione e l'uscita mantiene lo stato che aveva precedentemente (indicato appunto con Q_{n-1}).
- J=0, K=1: è lo stato di reset in cui l'uscita vale zero indipendentemente dal valore che essa aveva nell'istante precedente.
- J=1, K=0: è lo stato di set. L'uscita è a livello logico alto indipendentemente dal valore che essa aveva nell'istante precedente.
- J=1, K=1: stato di inversione. L'uscita commuta verso il valore complementare ($\overline{Q}_{n-1}$).

I FF sono dunque in grado di "ricordarsi" il precedente stato della propria uscita e, per questo motivo, sono impiegati come cella base di memorizzazione nelle memorie dei computer.

L'ingresso CP è detto Clock e consente di abilitare il FF al suo normale funzionamento. A seconda del simbolo usato per questo ingresso, si intende un diverso comportamento del FF rispetto al clock. Le possibilità sono le seguenti (i relativi simboli sono riportati nella figura 3):

1. High-Level sensitive: il FF funziona regolarmente se CP è a livello logico alto (+5V).

J	K	Q_n
0	0	Q_{n-1}
0	1	0
1	0	1
1	1	$\overline{Q}_{n-1}$

Figura 2: *Un Flip-Flop JK e la relativa tabella di verità*

2 Low-Level sensitive: il FF funziona regolarmente se CP è a livello logico basso (0V).
3 Positive Edge Triggered: il FF funziona solo nell'istante in cui CP commuta da livello basso a livello alto.
4 Negative Edge Triggered: il FF funziona solo nell'istante in cui CP commuta da livello alto a livello basso.

Il vantaggio di usare FF Edge Triggered consiste nel fatto che in questo modo è possibile "fotografare" lo stato degli ingressi nell'istante di commutazione del segnale di clock.

Si noti che nella realizzazione di figura 1 sono stati impiegati Flip-Flop Negative Edge Triggered.

FLIP-FLOP DI TIPO D

Osservando la tabella di verità di figura 2, si nota che se vincoliamo i due ingressi ad assumere valori opposti ed usiamo J come unico ingresso, otteniamo un dispositivo completamente trasparente detto Flip-Flop di tipo D (D-Latch). Per fare in modo che J e K assumano valori diversi tra loro, si inserisce una porta NOT tra i due ingressi. In questo modo otteniamo il FF di tipo D come illustrato in figura 4 (il FF di figura 4 è un Negative Edge Triggered). Ovviamente anche per i FF di tipo D si hanno le varie possibilità relative al segnale di clock già viste precedentemente.
La domanda nasce spontanea: ma che utilità può avere un dispositivo totalmente trasparente al segnale di ingresso? Non dimentichiamoci del segnale di clock! È vero infatti che il dispositivo è trasparente al segnale di ingresso, ma solo quando abilitato dal segnale di clock. Questo risulta molto utile in tutti quei casi in cui i dati di ingresso devono essere letti da altri dispositivi (ad esempio un microprocessore): il dato arriva all'ingresso D e quando arriva il segnale di clock passa inalterato attraverso il FF raggiungendo il dispositivo successivo. Tra un ciclo di clock a il seguente, l'ingresso D può anche variare più volte, ma le sue variazioni non saranno percepite a valle in quanto l'uscita Q rimane inalterata. A dimostrazione di ciò si consideri la figura 5 relativa ad un D-latch Negative Edge Triggered. L'uscita (in verde) assume il valore dell'ingresso (in marrone) solamente in corrispondenza dei fronti di discesa del segnale di clock CP (contrassegnati dalle frecce nere).

Figura 3: I vari simboli per indicare il comportamento di un FF rispetto al clock

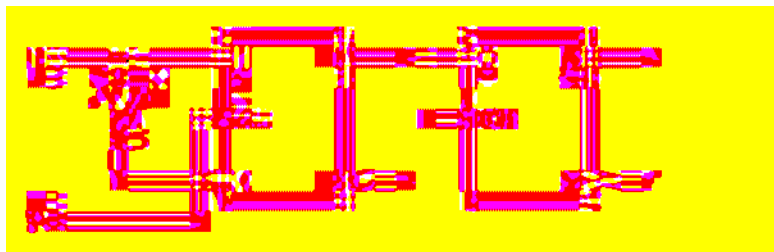



Figura 4: Un FF di tipo D ottenuto mediante un FF JK ed il relativo simbolo

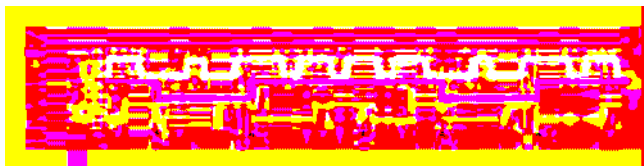



Figura 5: Segnali relativi ad un FF di tipo D: l'uscita assume il valore dell'ingresso solamente in corrispondenza dei fronti di discesa del segnale di clock.

FLIP-FLOP DI TIPO T

Che cosa accadrebbe se nel FF JK di figura 2 collegassimo i due ingressi J e K a livello logico alto in modo permanente? La risposta può essere trovata nella tabella di verità di figura 2 osservando l'ultima linea.
Quello che succede è che, ad ogni colpo di clock, l'uscita cambia continuamente stato. Questa particolare configurazione del FF JK, è detta Flip-Flop di tipo T (Toggle) ed il suo simbolo è riportato in figura 6.

Un Flip-Flop di tipo T è quindi un dispositivo che riceve in ingresso il solo segnale di clock e fornisce in uscita un segnale che commuta

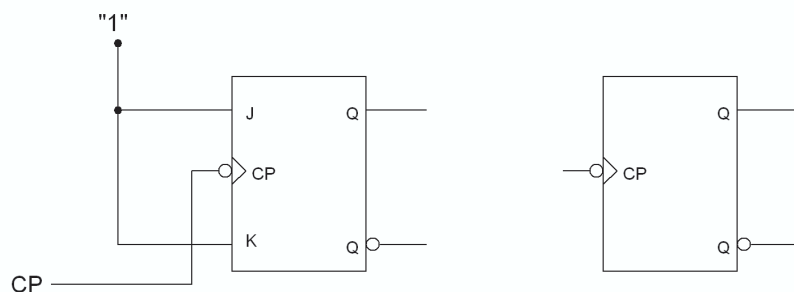



Figura 6: Un FF di tipo T ottenuto mediante un FF JK ed il relativo simbolo

Figura 7: Segnali relativi ad un FF di tipo T: la frequenza dell'onda in uscita è la metà di quella dell'onda in ingresso

continuamente tra i due stati logici ad ogni colpo di clock. Anche un dispositivo di questo tipo potrebbe sembrare inutile, ma se osserviamo il diagramma temporale di figura 7, ottenuto applicando un'onda quadra al FF di figura 6, notiamo una particolarità nei due segnali. Poiché il FF utilizzato è un Negative Edge Triggered, la commutazione dell'uscita si ha solo in corrispondenza del fronte di discesa del segnale CP, per cui il risultato è un'onda quadra la cui frequenza è la metà della frequenza del segnale di ingresso.

UN CONTATORE CON FLIP-FLOP DI TIPO T

Se osserviamo il circuito di figura 1 è possibile notare che i due FF sono connessi in modalità Toggle, dato che i loro ingressi sono connessi al positivo di alimentazione. Per comodità, l'interconnessione tra i due FF è riportata anche in figura 8. La particolarità del circuito sta nel fatto che l'ingresso CP del FF1 è pilotato manualmente mediante il pulsante SW, mentre il CP del FF2 è pilotato dall'uscita Q1 del primo. Alla pressione del pulsante si connette a massa l'ingresso CP di FF1 provocando la commutazione della sua uscita. L'uscita di FF2 commuta invece solamente quando l'uscita del primo ha una commutazione dal livello logico alto a quello basso.

Partendo dalla condizione in cui le uscite dei due FF sono azzerate (condizione peraltro garantita dalla presenza della rete R1, C1) e supponendo di premere ripetutamente SW (applicando quindi un'onda quadra all'ingresso CP del primo FF), si ottengono i segnali riportati in figura 9.

Poiché quando un'uscita è a livello alto, il relativo relè è eccitato (quindi la lampada è accesa) è possibile vedere, come riassunto in figura 9, che la sequenza di accensione delle lampade è quella dettata dalle specifiche richieste.
La connessone mostrata in figura 8 realizza un contatore binario a due bit, in grado cioè di contare gli impulsi in ingresso utilizzando

Figura 8: I Flip-Flop della soluzione proposta

Figura 9: I segnali ottenuti premendo ripetutamente il pulsante SW

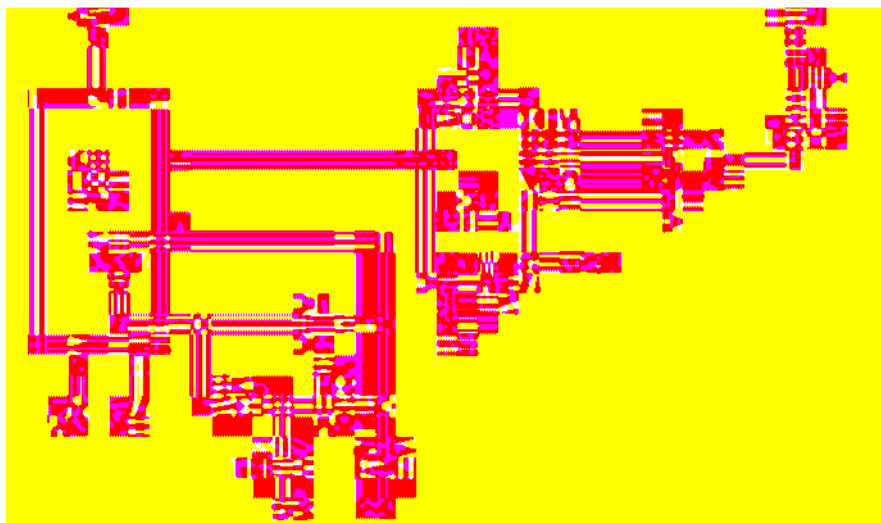



Figura 10: La risposta al quesito del mese scorso

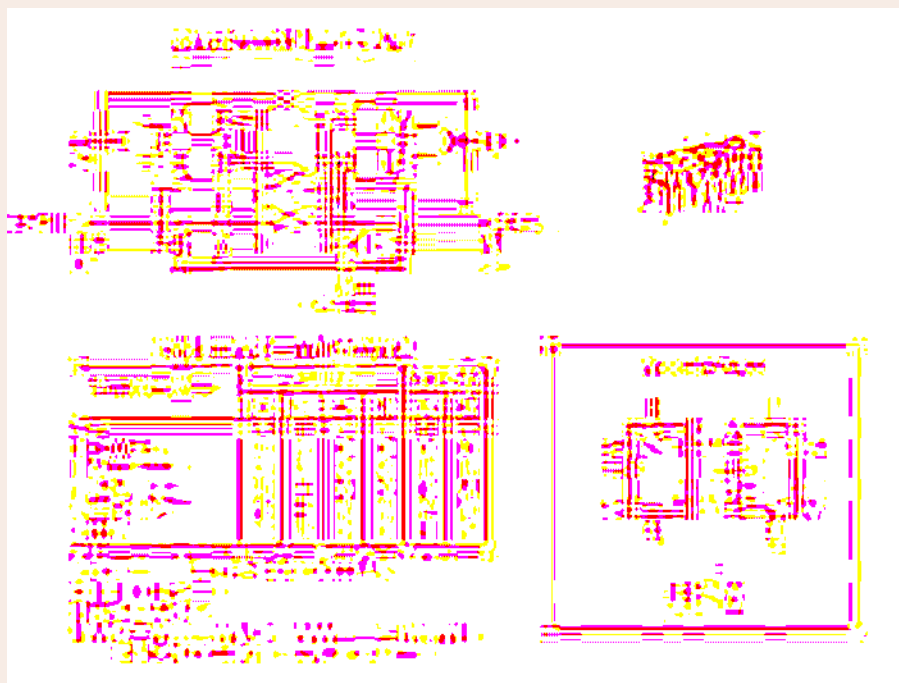
il codice binario. Il conteggio in binario a due bit va da zero a tre e comporta la seguente sequenza: 0=00, 1=01, 2=10, 3=11. Ovviamente è possibile espandere il conteggio aggiungendo altri flip-flop di tipo T.

IL QUESITO DEL MESE

Come modifichereste il circuito di figura 1 per consentire la gestione di tre lampade? Questo è il quesito che propongo questo mese. Non sarà troppo difficile se avete capito il funzionamento di un contatore con FF di tipo T!

Risolviamo ora il questo relativo agli oscillatori al quarzo proposto il mese scorso. IC2 effettua il conteggio dei secondi trascorsi, quindi dopo 10 secondi le sue uscite Q4÷Q1 assumeranno i valori 1010 (il corrispondente binario del numero 10). Si tratta allora di riconoscere tale combinazione delle uscite per azionare il LED. Il modo più semplice è quello di impiegare una porta NAND collegata come in figura 10. Solo in corrispondenza della combinazione 1010 ai suoi ingressi l'uscita sarà a livello basso ed il diodo LED risulterà acceso. Il LED resterà acceso per un secondo e, qualora si volesse aumentare il tempo di accensione, sarà necessario inserire un multivibratore monostabile innescato dalla porta NAND. Vorrei porvi un ulteriore quesito… secondo voi perché ho proposto di utilizzare una porta NAND e non una porta AND? La risposta è nascosta tra le righe della rubrica "Praticamente" pubblicata sul n.225 di Marzo 2004.

74LS112 - DUAL JK NEGATIVE EDGE TRIGGERED FLIP-FLOP

NOVITÁ

Imparare a progettare... ...progettando!

ELETTRONICANDO
vi aspetta su FareELETTRONICA
a partire da GENNAIO 2005

Esistono molti libri di testo per imparare l'elettronica, ma spesso, dopo le prime pagine, le formule, le tabelle, i valori, sembrano divenire qualcosa di incomprensibile.

Per questo motivo Fare Elettronica ha pensato di presentare il nuovo corso a puntate di elettronica analogica: "ELETTRONICANDO".

Senza la pretesa di rimpiazzare i libri di testo, "ELETTRONICANDO" presenta i principali argomenti di elettronica analogica in modo semplice e pratico, affinché siano comprensibili da tutti. Se poi pensi di non avere gli strumenti di misura per provare i circuiti... niente paura, "ELETTRONICANDO" ti insegnerà ad usare i simulatori software per verificare i risultati teorici.

Per non perdere nessuna puntata del corso, **abbonati subito!**

INDICE DELLE USCITE

PRIMA PARTE: (FE-235)
1. Introduzione
2. Un po' di storia
3. Gli strumenti necessari

SECONDA PARTE (FE-236)
4. Materiali conduttori, semiconduttori ed isolanti
5. I circuiti elettrici
6. Componenti dei circuiti elettrici: dispositivi passivi ed attivi
7. I generatori elettrici
8. Il resistore
9. Il condensatore
10. L' induttore

TERZA PARTE (FE-237)
11. Semplici circuiti con resistori
12. Carica e scarica di un condensatore
13. Cenni ai filtri passivi e relative risposte in frequenza (come leggere un diagramma di Bode)

QUARTA PARTE (FE-238)
14. Il semiconduttore intrinseco
15. Il semiconduttore estrinseco: drogaggio di tipo p ed n
16. La giunzione pn
17. Polarizzazione della giunzione pn
18. La caratteristica tensione-corrente del diodo pn

QUINTA PARTE (FE-239)
19. Un semplice circuito col diodo pn: il rettificatore
20. Il diodo zener
21. Il regolatore a diodo zener
22. Cenni ad altri tipi di diodi (varicap, LED, Schottky)

SESTA PARTE (FE-240)
23. Il transistor bipolare (struttura, caratteristiche e zone di funzionamento)
24. Amplificatori con bjt: base comune, collettore comune ed emettore comune.

SETTIMA PARTE (FE-241/242)
25. Cenni al MOSFET (struttura, caratteristiche e tipi)
26. Cenni al JFET (struttura, caratteristiche e tipi)

OTTAVA PARTE (FE-241/242)
27. Amplificatore operazionale (caratteristiche e parametri tipici)
28. Configurazione invertente e non invertente
29. Inseguitore di tensione

NONA PARTE (FE-243)
30. Circuiti con amplificatori operazionali: convertitore corrente tensione
31. Circuiti con amplificatori operazionali: Stabilizzatore di tensione
32. Circuiti con amplificatori operazionali: amplificatore differenziale

DECIMA PARTE (FE-244)
33. Timer 555: struttura e caratteristiche
34. Configurazione astabile
35. Configurazione monostabile

UNDICESIMA PARTE (FE-245)
36. Circuiti in reazione
37. Oscillatore a ponte di Wien
38. Oscillatore a rete di sfasamento

DODICESIMA PARTE (FE-246)
39. Gli strumenti di misura: il multimetro
40. Gli strumenti di misura: l'oscilloscopio
41. Tabelle varie (codice colori, valori commerciali dei resistori, ecc...)
42. Pin-out dei più comuni circuiti integrati (555, operazionali, filtri, ecc.)

GLI STABILIZZATORI DI TENSIONE

settima parte

I REGOLATORI A COMMUTAZIONE CAPACITIVA

di Nico Grilloni
nicogrilloni@tin.it

Una recente serie di circuiti integrati regolatori, realizzata con tecnologia CMOS, lavora in commutazione senza ricorrere ad alcuna induttanza.

CONVERTITORI A COMMUTAZIONE CAPACITIVA

I convertitori a commutazione capacitiva sono regolatori integrati che consentono di realizzare stadi step-up, step-down e inverter con un limitatissimo numero di componenti esterni e, considerazione ancora più rilevante, senza la necessità di fare uso di induttori.
Quest'ultima caratteristica assume grande importanza per il tecnico progettista che, nella realizzazione degli stabilizzatori di tipo switching, trova il punto dolente proprio nell'induttore che, nella stragrande maggioranza dei casi dev'essere appositamente costruito. Il suo valore, infatti, desunto dai calcoli di progetto, lo rende quasi sempre di difficile reperibilità.
Per le loro minime dimensioni, i convertitori a commutazione capacitiva sono da considerarsi componenti elettivi per tutte quelle apparecchiature che, alimentate a batteria, hanno particolari esigenze di costo e ingombro.
Il dimensionamento in pratica non esiste dal momento che, per il loro corretto funzionamento, sono sufficienti solo alcuni condensatori il cui valore è generalmente fornito dal costruttore. Molto più importante è invece il cablaggio e la scelta degli stessi condensatori che dovranno essere elettrolitici al tantalio o ceramici, o, meglio ancora, elettrolitici ai polimeri di alluminio. A questo proposito si raccomanda a Lettore che dovesse realizzare questi regolatori, di eseguire cablaggi di dimensioni estremamente contenute e di attenersi alle indicazioni qui, in proposito, riportate. Utilizzare un tipo di condensatore al posto di un altro anche se di eguale capacità, non significa pervenire necessariamente allo stesso risultato.

SCHEMI DI PRINCIPIO DEI REGOLATORI A COMMUTAZIONE CAPACITIVA

La figura 1 riporta la schematizzazione circuitale di un inverter in grado di fornire una tensione di uscita V_o di –5 V con una tensione

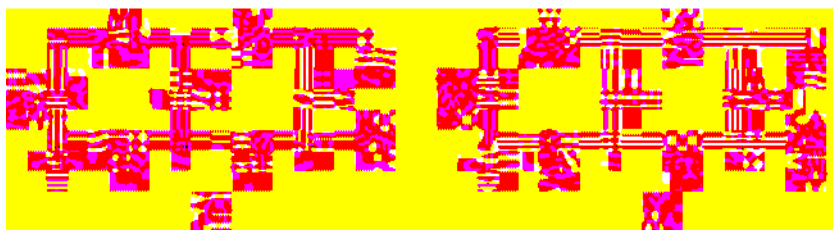



Figura 1: *Schematizzazione del funzionamento di un circuito integrato a commutazione capacitiva che fornisce una tensione $V_o = -V_i$*

V_i di ingresso di + 5 V. I deviatori $S_1 \div S_4$, sono ovviamente dei CMOS.
Il funzionamento è facilmente comprensibile se il ciclo di lavoro si scinde nella due fasi (a) e (b). Nella fase (a), nella quale i deviatori S_1 e S_2 sono chiusi e S_3 e S_4 aperti, la tensione di ingresso carica il condensatore C_1 alla tensione V_i; nella fase (b), nella quale i deviatori S_1 e S_2 sono aperti e S_3 e S_4 chiusi, il condensatore C_1 trasferisce la sua carica al condensatore C_2 ai cui capi si presenterà pertanto la stessa tensione V_i. Poiché le armature positive di entrambi i condensatori sono adesso a massa, il pin di uscita della V_o, connesso alle armature negative, si troverà al di sotto del livello di massa di una tensione pari alla tensione a cui si è caricato C_2, ovvero a $-V_i$, ossia a – 5 V.

Non molto diverso è il funzionamento degli integrati regolatori di tipo step-up, che duplicano la tensione di ingresso, dando quindi luogo a una $V_o = 2\ V_i$. La figura 2 ne riporta la schematizzazione. Nella fase (a) i deviatori S_1 e S_2 sono chiusi e quindi la tensione V_i va a caricare il condensatore C_1. Nella fase (b), nella quale S_1 e S_2 sono aperti e S_3 e S_4 sono chiusi, il condensatore C_1 trasferisce l'energia immagazzinata nel processo di carica al condensatore C_2 che, in parallelo con C_1, è praticamente in serie alla tensione V_i rispetto al pin di uscita della V_o. Il valore di questa sarà quindi pari alla somma della V_i e della tensione di carica di C_2 che è anch'essa pari a V_i. Sarà pertanto: $V_o = 2\ V_i$.

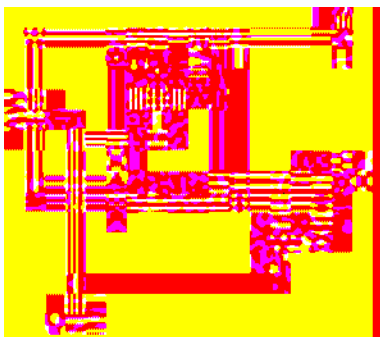



Figura 2: *Schematizzazione del funzionamento di un circuito integrato a commutazione capacitiva che, comportandosi da duplicatore, fornisce una tensione $V_o = 2\ V_i$*

L'INVERTER LM828

La figura 3 riporta uno stadio inverter realizzato con il regolatore LM828, integrato particolarmente indicato nella apparecchiature con alimentazione a batteria, come cellulari e strumenti di misura. Le dimensioni sono minime: misura infatti circa (3 mm x 1,6 mm) e l'altezza supera di poco il millimetro.
Come si può constatare sono sufficienti due componenti esterni – i condensatori elettrolitici al tantalio C_1 e C_2 da 10 µF – per realizzare un alimentatore in grado di fornire, con una tensione di ingresso V_i (+ 1,8 ÷ + 5,5) V, una tensione di uscita V_o (–1,8 V ÷ –5,5 V) con una corrente di 20 mA.
Disponendo in cascata due LM828 come illustra la figura 4, si ottiene un alimentatore duplicatore di tensione e invertente. Si ha cioè in uscita: $V_o = -10$ V per $V_i = 5$ V. Ponendo in cascata *n* regolatori, la tensione di uscita, in assenza di carico, sarà pari a $(-n \cdot V_i)$; ciò significa che disponendo in cascata tre LM828, supponendo, per esempio, una $V_i = 5$ V, si otterrà una $V_o = -15$ V; per quattro stadi in cascata, sempre nell'ipotesi di una $V_i = 5$ V, si avrà una $V_o = -20$ V, e così via. In ogni caso però, la corrente totale erogata dalle varie uscite non deve superare i 40 mA.
Ricorrendo alla configurazione

Figura 3: *Ponendo in circuito soltanto due condensatori, con l'LM828 si realizza un regolatore con funzione inverter*

Figura 4: *Ponendo in cascata due LM828 si realizza un duplicatore inverter, ossia un circuito regolatore che fornisce una tensione di uscita doppia della tensione di ingresso e di segno opposto*

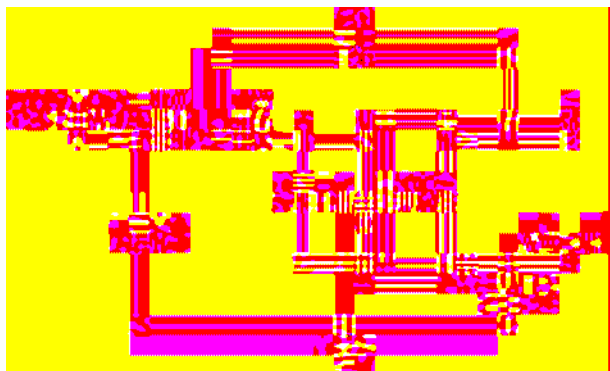



Figura 5: *Configurazione circuitale con l'LM828, che assolve contemporaneamente la funzione step-up ($V_{o1} = 2\ V_i$) e la funzione inverter ($V_{o2} = -V_i$)*

Figura 6: *Con un LM828 e un regolatore a bassa dropout come l'LP2980 si ottiene una tensione di uscita V_o perfettamente stabilizzata e, ove si ponga un trimmer al posto della R_1, anche variabile*

circuitale riportata nella figura 5, con un singolo LM828 si possono ottenere due diverse tensioni di uscita, una di tipo step-up dal momento che si ha $V_{o1} = 2 \cdot (V_i - V_F)$, dove V_F è la c. d. t. determinata dai diodi D_1 e D_2, e una di tipo inverter $V_{o2} = -V_i$.

Nel caso sia necessario avere una tensione di uscita perfettamente stabilizzata e variabile, si potrà fare ricorso alla configurazione riportata nelle figura 6 nella quale l'uscita dell'LM828 pilota il regolatore LP2980 di bassa dropout ($\Delta V = 120$ mV per $I_L = 50$ mA).
La variabilità della tensione di uscita si ottiene modificando il valore della resistenza R_1. In funzione della V_o voluta, si attribuisce alla R_1 un qualsiasi valore compreso fra 2,2 kΩ e 5 kΩ, e quindi si ricaverà R_2 dall'espressione:

$$V_o = [1{,}23 \cdot (R_1 + R_2)] / R_2$$

dove 1,23 è la tensione di riferimento V_{ref} dell'LM2980.
Dalla [5] si ha, infatti:

$$R_2 = (1{,}23 \cdot R_1) / (V_o - 1{,}23)$$

Volendo, per esempio, una tensione V_o di uscita pari a 12 V, data la bassa caduta di tensione input-output che presenta l'IC LM2980, sarà sufficiente applicare una tensione di ingresso anche solo di 11 V. Quindi attribuito alla R_1 il valore di 2,2 kΩ, si avrà per R_2:

$$R_2 = (1{,}23 \times 2200)/(12 - 1{,}23) = 251\ \Omega$$

STEP-UP E STEP-DOWN CON L'LM3352

La figura 7 a riporta il regolatore LM3352 che, previsto come regolatore per sistemi alimentati a batteria alcalina (a ioni di Litio o al NiCd o NiMH), consente di avere una V_o fissa e costante di 3,3 V con tensioni V_i di ingresso che possono assumere qualsiasi valore fra 2,5 V e 5,5 V. Ciò è evidenziato dal diagramma della

Figura 7a, 7b: *L'LM3352 (a) si comporta come regolatore step-up e step-down. In b si può notare come per una tensione V_i compresa fra 2,5 V e 5,5 V, l'uscita rimanga ancorata a 3,3 V*

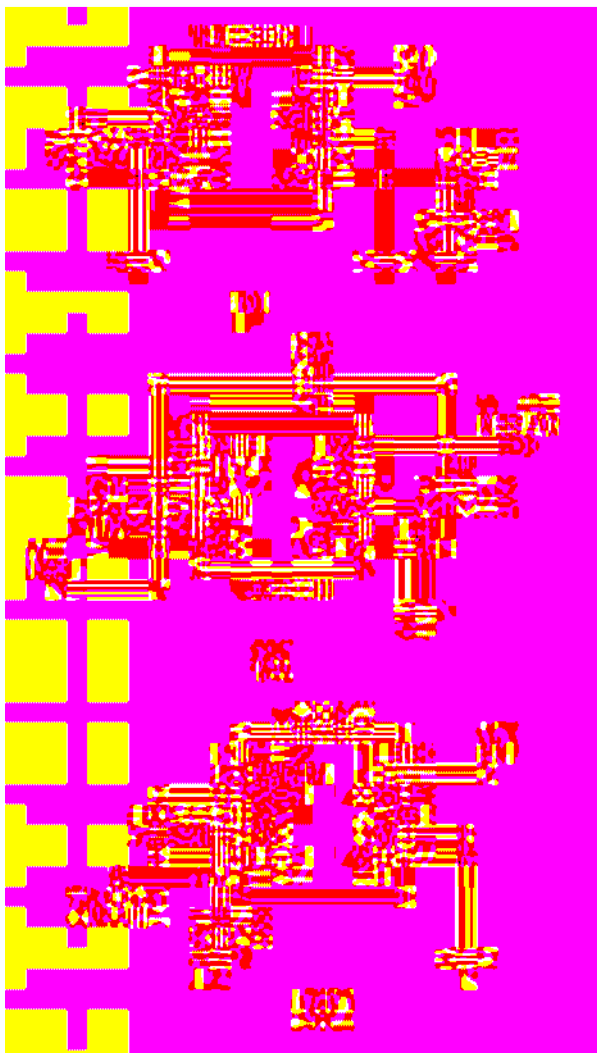



Figura 8a, 8b, 8c: *Stadi regolatori con LM2660/1 in grado di fornire una tensione $V_o = -V_i$ (a) – configurazione inverting, o $V_o = 2\ V_i$ (b) – configurazione da duplicatore, o $V_o = V_i / 2$ (c) – configurazione da divisore*

figura 7 *b* che indica, per l'appunto, come per qualsiasi valore della tensione V_i di ingresso compreso fra 2,5 V e 5,5 V, la tensione di uscita rimanga ancorata a 3,3 V. L'LM3352, che si comporta quindi come regolatore step-up e step-down, è fornito dalla National Semiconductor per tensioni di uscita di 2,5 V, 3,0 V e 3,3 V, ma, a richiesta, è disponibile con tensioni di uscita da 1,8 V a 4,0 V con incrementi, all'interno di questa gamma, di 0,1 V. Con un carico di 50 mA (ma si può arrivare fino a un massimo di 200 mA) e disponendo in uscita una capacità da 33 µF, il residuo in alternata è tipicamente di 75 mV, mentre la corrente di riposo e dell'ordine di 300 µA che si riducono a 5 µA all'intervento dello shutdown. L'LM3352 è dotato di protezione termica che, in condizioni gravose, ne interrompe il funzionamento, funzionamento che si autoripristina quando la temperatura della giunzione rientra nei limiti di sicurezza.

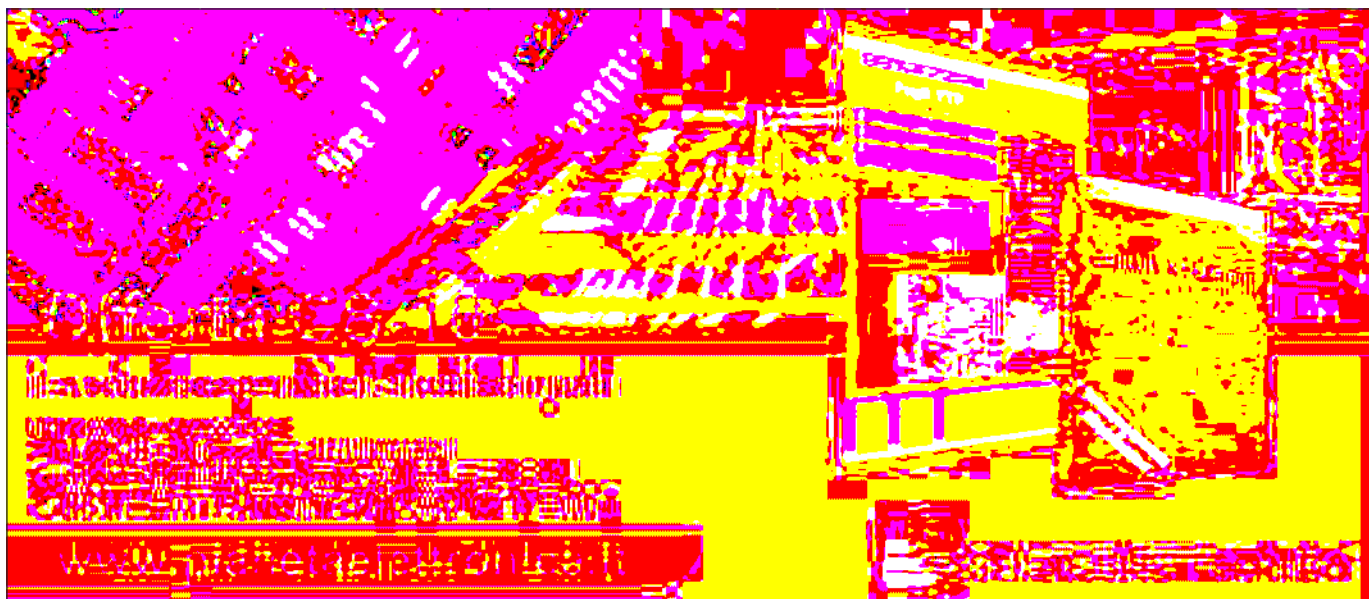



Figura 9: *Regolatore triplicatore: ponendo due LM2660 (o LM2661) in cascata, si ottiene, oltre a una uscita $V_o = 2\ V_i$, anche una tensione $V_o = 3\ V_i$*

Figura 10: *Regolatore che consente di ottenere da una tensione d'ingresso positiva, una tensione di uscita negativa e variabile*

INVERTER, DUPLICATORE E DIVISORE DI TENSIONE CON L'LM2660-1

La serie di questi circuiti integrati – prevista per computer portatili (laptop), telefoni cellulari, strumentazione medicale e strumenti di misura, alimentazione di op-amp, ecc. – consente sia di ottenere una tensione V_o negativa da una V_i positiva (*inverting*), sia di duplicare la V_i fornendo una $V_o = 2\ V_i$ (*voltage doubler*), sia di dimezzare la V_i dando in uscita una $V_o = V_i / 2$ (*splitting*). In ogni caso la serie è in grado di fornire una corrente massima di 100 mA con una corrente di riposo inferiore a 120 µA, e con un'efficienza del 90 %.

Le tre configurazioni corrispondenti sono riportate nelle figure 8 *a*, *b* e *c*. Si noti come nel primo e nel terzo circuito siano sufficienti soltanto due condensatori elettrolitici per il loro funzionamento, mentre la configurazione di duplicazione richieda, oltre ai due condensatori, un diodo.

La frequenza dell'oscillatore interno è fissata a 80 kHz nel modello LM2661, mentre nel modello LM2660 la medesima frequenza può assumere il valore di 10 kHz o di 80 kHz a secondo che il pin FC (Frequency Control) sia lasciato aperto o connesso al pin V+.

Nell'LM2661 il pin FC è sostituito dal pin SD (ShutDown) che, connesso normalmente a massa, può essere utilizzato per disattivare il regolatore portandolo al pin V+; in quest'ultima condizione la sua corrente di riposo va a 0,5 µA. Il pin Osc., connesso a un condensatore interno da 15 pF, consente il pilotaggio dei regolatori con un clock esterno la cui frequenza massima non può eccedere i 150 kHz; in questo caso il modo di connessione del pin 7 è indifferente. Ponendo, quindi, un condensatore fra il pin Osc. e massa, la frequenza di oscillazione può essere spostata verso il basso. Il costruttore – National Semiconductor – fornisce, a tal proposito, un diagramma con due curve, una per FC aperto e una per FC connesso al V+, che riporta la frequenza di oscillazione in funzione del valore di capacità attribuito al condensatore esterno.

A ulteriore dimostrazione della versatilità di questi circuiti integrati della più recente generazione, la figura 9 riporta un triplicatore di tensione ottenuto con due LM2660 (o LM2661) in cascata, mentre, infine, la figura 10 espone uno stadio regolatore che converte una tensione di ingresso positiva in una tensione di uscita negativa e variabile. Per questo risultato si è posto il regolatore a bassa dropout LP2951 (0,38 mV di caduta input-output con 100 mA di corrente erogata), in cascata con un LM2660.

Essendo:

$$V_o = V_{ref} \cdot [(R_1 + R_2) / R_2]$$

con V_{ref} = 1,235 V, tensione di riferimento caratteristica dell'LP2951, ponendo, per esempio R_2 = 3,3 kΩ e usando per R_1 un trimmer da 27 kΩ, si può ottenere la variabilità della tensione di uscita fra il valore V_{ref} = 1,235 V con R_1 in cortocircuito, e il valore di 11,34 V con R_1 completamente inserito.

INTRODUZIONE AI SISTEMI WIRELESS DIGITALI

terza parte

IL BLUETOOTH

di Sandro Romagnoli
s.romagnoli@farelettronica.com

Il Bluetooth è una tecnologia a basso costo e molto flessibile, che permette di "tagliare i fili" ad un grande numero di apparecchiature elettroniche e di aprire nuove possibilità. Vediamo a cosa serve e come si pone in relazione con gli altri standard di comunicazione wireless più diffusi, come Wi-Fi e GSM/GPRS.

L'idea è sempre quella: eliminare l'intrico di cavi di collegamento che caratterizza l'utilizzo di più apparecchiature elettroniche insieme, siano essi i vecchi cavi RS232, che troviamo ancora nei modem per la linea telefonica analogica o le nuove connessioni seriali ad alta velocità come USB e FireWire che vengono impiegate in dispositivi caratterizzati da una più elevata multimedialità, come fotocamere, scanner e stampanti.
Alla diffusione sempre maggiore che accompagna il PC e quindi tutte le sue periferiche, si aggiunge la rivoluzione dettata dall'utilizzo del telefono cellulare, che è diventato in pochi anni un vero e proprio computer palmare e richiede un livello elevato di connettività per scambiare dati con il PC stesso o con altri dispositivi cellulari.
Da questa premessa è nato lo sforzo delle più grandi aziende operanti nel settore della telefonia cellulare, fra cui Motorola, Nokia ed Ericsson, teso a definire una tecnologia di connessione wireless a basso costo che potesse risolvere tutte le nuove esigenze di connettività che si venivano a creare (la tecnologia IrDA a raggi infrarossi già usata da tempo ha il limite di funzionare solamente in assenza di ostacoli). Il risultato degli studi effettuati è il Bluetooth (www.bluetooth.com).
Le specifiche iniziali erano molto ambiziose soprattutto riguardo l'obbiettivo di costo sotto i 5 dollari per ogni radio, ma solo negli ultimi due anni l'industria microelettronica è

Figura 1: *Adapter Bluetooth USB per PC*

Figura 2: *Auricolare Bluetooth per telefoni cellulari*

avanzata al punto da riuscire a produrre dei singoli circuiti integrati CMOS in grado di ospitare al loro interno tutta la parte in banda base e quasi tutta la sezione RF (in genere rimane fuori solamente l'antenna) consentendo di abbassare drasticamente i costi e consentire così una grande diffusione. Oggi appaiono sul mercato continuamente nuovi dispositivi dotati di interfaccia Bluetooth, in primo luogo i telefoni cellulari, ma anche GPS, stampanti e modem: nasce così il concetto di PAN (Personal Area Network), una rete wireless a corto raggio formata da tanti dispositivi elettronici che si scambiano dati fra loro.
In figura 1 si vede un adattatore USB, mentre in figura 2 un auricolare.

PRINCIPI DI FUNZIONAMENTO

L'idea di progetto iniziale di realizzare un sistema che permettesse la trasmissione di voce e dati e che fosse semplicissima da usare per gli utenti finali, ha portato allo sviluppo di uno stack di protocollo veramente complesso. Uno dei fondamenti è che le radio Bluetooth devono essere in grado di riconoscersi e formare delle reti fra loro in modo autonomo senza praticamente l'intervento esterno, al contrario del Wi-Fi in cui è normalmente richiesta una seppur minima conoscenza di base delle reti di calcolatori.

Nella maggior parte degli adattatori Bluetooth per PC che si trovano in commercio, il protocollo è diviso in due parti (vedi figura 3, che ne riporta una versione semplificata): la parte di più basso livello (colore giallo) è ospitata sull'adattatore stesso, che può essere una chiavetta USB o una PC Card, mentre l'altra (colore azzurro) è implementata sul driver che gira sul PC, che è normalmente fornito dal costruttore dell'adattatore stesso. Per far parlare fra loro le due parti dello stack è definito un livello di interfaccia chiamato **HCI** (Host Control Interface). Chi è abituato a lavorare con interfacce semplici tipo la RS232, sa che lo standard definisce solamente la parte più vicina all'hardware, mentre non dice niente su come impacchettare o frammentare i dati che vogliamo inviare o ricevere su quell'interfaccia. Al contrario Bluetooth introduce il concetto di profilo che arriva quasi ad un livello applicativo e consente ad esempio di trasferire un file fra un nodo Bluetooth ed un'altro.

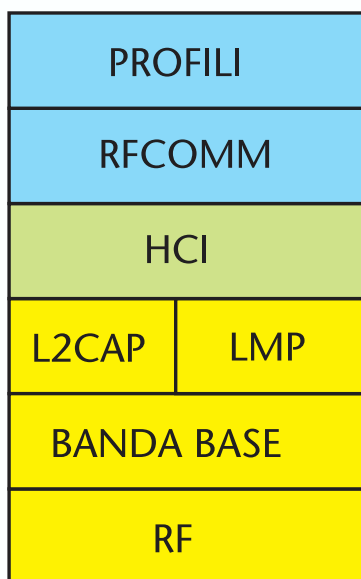



Figura 3: *Stack Bluetooth*

Livello RF

A questo livello sono definite le caratteristiche del segnale RF utilizzato dalle radio Bluetooth. La banda di lavoro è quella ISM posta a 2,4GHz (la stessa del Wi-Fi), che viene divisa in 79 canali, ognuno dei quali è ampio 1 MHz e, per far fronte alle interferenze, il sistema adotta un **FHSS** (Frequency Hopping Spread Spectrum) con 1600 hops/secondo, cioè 1600 volte in un secondo si "salta" su di un diverso canale fra i 79 disponibili. La modulazione utilizzata è la **GFSK** (Gaussian Frequency Shift Keying) che è una variante della modulazione di frequenza che invece di usare un'onda quadra per rappresentare i bit da trasmettere, adotta una forma d'onda che ricorda una campana di Gauss, ottenendo così una maggiore efficienza spettrale. Per quanto riguarda la potenza del segnale RF emesso, i dispositivi in commercio si riferiscono in genere alla portata in metri (ovviamente quello specificato è puramente indicativo) e se ne trovano da 10 metri (potenza dell'ordine di 1mW) o da 100 metri (potenza dell'ordine di 100mW).

Banda Base

Il funzionamento in Frequency Hopping, fa sì che la più piccola unità di informazione disponibile sia lo *Slot*, che consiste nella sequenza di bit trasmessa mentre la radio è posizionata su un canale. Poiché si lavora a 1600 hops/s, uno slot dura 625 microsecondi. La comunicazione avviene dividendo i messag-

gi in pacchetti ed un pacchetto può occupare uno o più slot (1, 3 o 5): mentre con l'utilizzo di un solo slot si può arrivare ad una velocità di trasferimento di 172 Kbit/s, l'utilizzo di più slot consente di aumentare la velocità, arrivando ad un massimo teorico di 721 Kbit/s utilizzandone 5.

Livello L2CAP
L2CAP sta per Logical Link Control and Adaptation Protocol, qui vengono svolti il multiplexing dei dati provenienti dai protocolli di livello superiore (cioè, una unica radio Bluetooth può essere usata da diversi applicativi ad alto livello, ognuno dei quali usa un certo modo per trattare le informazioni), la frammentazione, il riassemblaggio dei pacchetti ed il trasporto delle informazioni di "qualità del servizio" (QoS) che servono quando vengono impiegati dei flussi di dati in tempo reale, come ad esempio l'audio o il video che non tollerano ritardi di trasmissione superiori ad una certa soglia.

Livello LMP
LMP vuol dire *Link Manager Protocol* e si occupa della formazione e della gestione di reti fra i diversi nodi Bluetooth.

Livello HCI
Host Control Interface fornisce una interfaccia standard ai diversi hardware che implementano una radio Bluetooth, in modo che un medesimo driver scritto per funzionare su un PC possa funzionare con tutti gli adattatori presenti sul mercato, indipendentemente dal costruttore dell'adattatore stesso.

Livello RFCOMM
Questo livello utilizza la radio Bluetooth per emulare un collegamento seriale standard (tipo RS232), viene utilizzato da molti profili di alto livello ad esempio per trasferire files.

Profili
I profili rappresentano l'interfaccia tramite la quale un utente interagisce con i dispositivi Bluetooth. I profili disponibili sono tanti e ne vengono continuamente introdotti dei nuovi per gestire al meglio le nuove periferiche che vengono immesse sul mercato.
Tipicamente, gli adattatori Bluetooth per PC supportano un ampio spettro di profili, mentre invece i dispositivi più semplici, come ad esempio i telefoni cellulari ne supportano un sottoinsieme. Alcuni esempi sono:

- **Serial Port Profile** (SPP): come si intuisce dal nome, questo profilo implementa una porta seriale virtuale. Consideriamo due PC dotati entrambi di radio Bluetooth: se il driver supporta questo profilo, avranno a disposizione delle porte COM virtuali che possono essere utilizzate mediante i normali programmi tipo HyperTerminal di Windows. Aprendo sui due PC due sessioni di terminali sulle rispettive porte virtuali potranno scambiarsi dati esattamente come se fossero collegati mediante un cavo null-modem RS232 attraverso la radio Bluetooth.
- **Object Exchange** (OBEX): consente di trasferire files fra dispositivi dotati di radio Bluetooth, siano essi due PC, oppure un PC ed una fotocamera digitale.
- **Dial Up Network** (DUN): questo profilo è in genere implementato sui modem Bluetooth e consente ad un PC con Bluetooth di utilizzarlo come se fosse collegato con un cavo. Questo profilo è in genere implementato nei telefoni cellulari, che possono funzionare da modem per il PC.
- **Headset** (Cuffia): è usato anch'esso nei cellulari e consente l'utilizzo dell'auricolare wireless.
- **Local Area Network** (LAN): tramite questo profilo è possibile accedere ad una rete locale TCP/IP in modo wireless.
  Consideriamo un PC collegato ad una rete Ethernet e dotato anche di una interfaccia Bluetooth: questo può diventare un punto di accesso wireless alla rete per tutti i dispositivi mobili Bluetooth che dispongono di questo profilo, ad esempio computer palmari.
- **Personal Information Manager** (PIM): serve per scambiare fra dispositivi Bluetooth informazioni quali biglietti da visita elettronici, e-mail, note o appuntamenti.
- **Human Interface Device** (HID): consente di connettere ad un PC periferiche come mouse e tastiera.

Per poter utilizzare un profilo in

una connessione fra due nodi, il profilo stesso deve essere supportato esplicitamente da entrambi, altrimenti la connessione non può avere luogo. Esiste un protocollo apposito chiamato **SDP** (Service Discovery Protocol) che ha proprio la funzione di scoprire quali sono i profili supportati da un nodo Bluetooth remoto.

STRUTTURA DI UNA RETE

Come abbiamo detto in precedenza, le radio Bluetooth sono in grado autonomamente di formare una rete fra loro e per fare questo adottano una strategia di tipo *Master/Slave*. Il Master è il nodo della rete che parte per primo e che coordina il funzionamento e la sincronizzazione degli altri elementi: gli Slave. Normalmente le radio Bluetooth sono in grado di funzionare sia come Master che

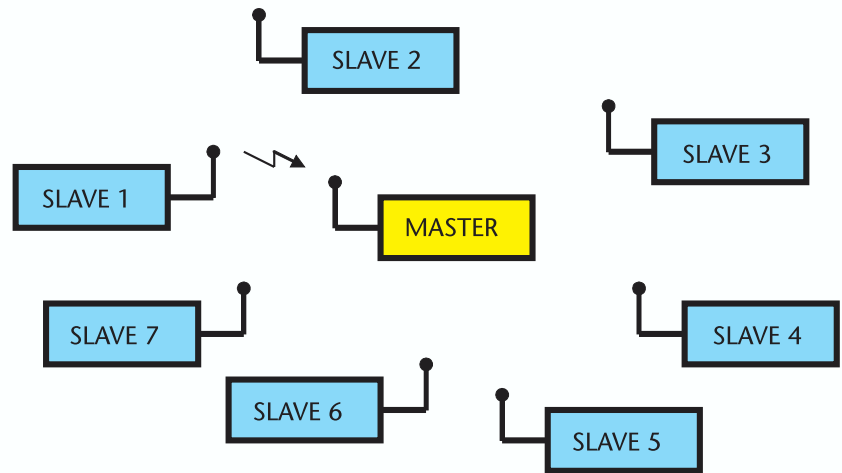



Figura 4: *Piconet: la più semplice tipologia di rete Bluetooth*

come Slave. La più semplice topologia di rete Bluetooth è chiamata *Piconet* ed è formata da un Master e fino ad un massimo di sette Slave (figura 4).

Ogni radio Bluetooth è dotata di un indirizzo unico al mondo costituito da 48 bit (ad esempio 00:0B:0D:21:0C:84) ed il Master della Piconet parte da questo valore per determinare la sequenza di salti fra i 79 canali disponibili che viene adottata anche dagli Slave.
Perché si possa formare una Piconet, il Master comincia ad inviare dei pacchetti in cui interroga gli eventuali altri dispositivi Bluetooth presenti nell'area di copertura (questa fase è detta *Inquiry*), che intanto stanno in ascolto per un po' di tempo sui diversi canali in attesa di essere trovati. Una volta che uno Slave sente il pacchetto inviato dal Master gli risponde inviando il proprio indirizzo e così nel giro di pochi secondi il Master è a conoscenza di tutti i dispositivi Bluetooth presenti e con cui può creare una Piconet.
Una volta creata la Piconet, ad ogni nodo viene assegnato un indirizzo temporaneo da 1 a 7 (0 è il Master) tramite il quale i nodi possono parlare tra loro. Diverse Piconet possono convivere nella stessa area grazie al fatto che tutti i membri della stessa Piconet utilizzano una certa sequenza di salti fra i 79 canali disponibili (dettata dal nodo Master) che sarà diversa da Piconet a Piconet, certo, potrà succedere che occasionalmente più nodi si trovino a trasmettere contemporaneamente sullo stesso canale e questo potrà portare alla ritrasmissione della informazione e quindi ad una diminuzione della velocità media di trasferimento.
Esiste anche un tipo più ampio di rete chiamato Scatternet in cui più Piconet sono messe in comunicazione tra loro attraverso alcuni nodi comuni a più Piconet che si occupano di trasferire i pacchetti da una Piconet all'altra.
Per i dispositivi Bluetooth a batteria, sono previste delle configurazioni di "riposo" ad assorbimento di corrente ridotto, che possono essere utilizzate nei momenti in cui non viene richiesto uno scambio dati (modalità *Park* e *Sniff*).

DIVERSI TIPI DI COLLEGAMENTO

Uno dei punti di forza del Bluetooth è la grande flessibilità che gli consente di trasportare sia voce che dati: questo è infatti stato richiesto fin dalle specifiche iniziali del progetto al contrario di quanto è avvenuto col Wi-Fi che è nato per gestire unicamente traffico dati e che richiede l'ulteriore introduzione di protocolli che gli consentano di supportare applicazioni multimediali in tempo reale. Infatti i requisiti richiesti dal traffico dati e da quello audio sono completamente diversi:

- **Dati**: quando si attua un trasferimento dati è di fondamentale importanza che il messaggio ricevuto sia uguale a quello trasmesso fino all'ultimo bit (si pensi al trasferimento di un file contenente un programma eseguibile). Vengono quindi previsti meccanismi di *Forward Error Correction* (FEC) in cui vengono trasmessi dei bit di controllo aggiuntivi al messaggio vero e proprio che consentono al ricevitore di ripristinare eventuali bit sbagliati ed anche di ritrasmissione dei blocchi di dati non ricevuti correttamente.
  Questo fa sì che possa essere variabile l'intervallo di tempo che intercorre fra la ricezione dei vari pacchetti dati che appartengono ad un medesimo messaggio.
- **Audio**: un flusso audio è costituito a partire da una sorgente analogica (normalmente un microfono), che viene poi digitalizzata e compressa mediante un opportuno algoritmo. Il flusso di bit risultante dalla compressione viene poi suddiviso in pacchetti che vengono inviati via radio. Il ricevitore processa i pacchetti che gli arrivano, ricostruendo il flusso di bit non compresso, ed attraverso un convertitore digitale/analogico va a comandare un altoparlante.
  Perché un collegamento di questo tipo mantenga una buona qualità è necessario che i pacchetti in cui è suddiviso il messaggio non subiscano un ritardo troppo grande tra uno e l'altro che porterebbe a dei "buchi" nell'ascolto, ossia, bisogna introdurre il concetto di *Qualità del Servizio*.

Bluetooth ha introdotto due

diversi tipi di collegamento fra i nodi all'interno di una Piconet, normalmente questi collegamenti vengono attivati in modo trasparente all'utente:

- **Asynchronous Connection Less** (ACL): questo è il collegamento normalmente utilizzato per il trasferimento dei dati. Connection Less significa che non è necessario stabilire una connessione logica fra due nodi prima di cominciare un trasferimento, ma ogni nodo può trasmettere un pacchetto ad un altro della Piconet in qualsiasi momento e poi si aspetta un pacchetto che attesta l'effettiva ricezione (Acknowledge ACK). Se il pacchetto ACK non viene ricevuto entro un tempo prestabilito il pacchetto viene ritrasmesso.
- **Synchronous Connection Oriented** (SCO): al contrario del collegamento precedente, qui occorre stabilire una connessione logica fra i due nodi interessati e fatto questo viene garantita una certa "qualità di servizio" riservando alcuni slot solamente per questo tipo di traffico che viene usato per la comunicazione audio. In questo modo viene garantito che la distanza temporale fra due pacchetti non supererà mai una data soglia.

SICUREZZA DEL COLLEGAMENTO

Con il termine sicurezza si intende la protezione dei dati che vengono scambiati via radio nei confronti di malintenzionati che se ne vogliono appropriare. Abbiamo infatti detto che le radio Bluetooth sono in grado di riconoscersi e formare spontaneamente una rete, ma come impedire questo se non lo vogliamo?
Innanzitutto, per creare una Piconet, è necessario che gli Slave rispondano alle richieste di identificazione del Master e molti dispositivi commerciali consentono di stabilire se il dispositivo stesso è "scopribile" o meno dagli altri. Quindi è necessario assicurarsi che questa modalità sia normalmente disabilitata se non vogliamo che gli altri "ci vedano", per poi abilitarla esplicitamente solamente quando è necessario.

Il Pairing

Un altro concetto molto importante a riguardo della sicurezza del Bluetooth è il concetto di Pairing (in inglese "appaiamento"): quando il Master (supponiamo che sia un PC con radio Bluetooth) ha effettuato la scansione per individuare gli altri nodi Bluetooth sotto copertura, può decidere di stabilire il collegamento con uno di questi. A questo punto verrà richiesto all'utente del PC di inserire un PIN code, cioè una specie di password formata da quattro cifre. Per stabilire la connessione Piconet anche all'utente del nodo Slave (che potrebbe essere un telefono cellulare) dovrà inserire un PIN code, che dovrà essere uguale a quello inserito sul PC. Una volta effettuata questa procedura i due nodi Bluetooth, PC e cellulare, si ricorderanno di essere stati appaiati e non chiederanno più esplicitamente l'inserimento del PIN code agli utenti.

Criptaggio dei dati

Oltre al Pairing, che consente di evitare la creazione di connessioni indesiderate, il Bluetooth prevede la possibilità di criptare i messaggi scambiati via radio utilizzando una chiave lunga fino a 128 bit. E' chiaro che le contromisure di sicurezza da adottare devono aumentare con l'importanza dei dati stessi che vogliamo scambiare.

BLUETOOTH E LE ALTRE TECNOLOGIE

Bluetooth va a definire un nuovo concetto di rete wireless (la Private Area Network) che si aggiunge a quelli già esistenti (la Wide Area network WAN e la Local Area network LAN) che consente di avere collegamenti senza fili fra un computer e le due periferiche, ma grazie alla sua flessibilità è in grado di fare da interfaccia fra le altre tecnologie radio disponibili.

Reti GSM/GPRS

Tutti ormai siamo utenti della rete GSM per quanto riguarda il traffico voce e ne sfruttiamo anche le possibilità di trasferimento dati mediante gli SMS. L'evoluzione del GSM rivolto al trasporto dati è il GPRS, che pur utilizzando le infrastrutture radio preesistenti, supporta il protocollo TCP/IP e consente di trasferire dati con velocità che arrivano mediamente a 30Kbit/s (contro i 9,6Kbit/s del GSM). Lo step successivo che

porterà la velocità media di trasferimento a diverse centinaia di Kbit/s è la terza generazione di telefonia cellulare, l'UMTS. Quando la rete UMTS si sarà diffusa, darà la possibilità ai suoi utenti di accedere ad Internet o di scambiarsi contenuti multimediali ovunque si trovino.

Reti Wi-Fi

Questi sistemi si stanno diffondendo soprattutto in ambito indoor per fornire accesso wireless ad internet a banda larga, con velocità che oggi raggiungono anche i 100 megabit/s. La grossa differenza rispetto alle reti cellulari è che lavorano nella banda ISM e quindi non necessitano di licenze plurimiliardarie. Sono in corso progetti per ottimizzare questi sistemi in modo da poter trasportare la voce (c'è chi ha pensato alle cabine telefoniche wireless via Wi-Fi), rendendoli così preferibili in alcuni ambiti alla tecnologia UMTS, soprattutto perché contraddistinta da costi di gestione ed installazione notevolmente inferiori.

La connessione wireless globale

Lo scenario futuro più ottimistico è quello che prevede una integrazione delle diverse tecnologie in modo da permetterci di poter usare di momento in momento quella migliore. Supponiamo che esista un telefono cellulare che supporti contemporaneamente le tecnologie Bluetooth, Wi-Fi ed UMTS: quando siamo in ufficio il telefonino si aggancia automaticamente alla rete Wi-Fi aziendale, consentendoci di telefonare attraverso il Voice over IP (VoIP) che trasporta l'audio utilizzando una normale rete TCP/IP. Quando poi usciamo dall'ufficio, il telefono si aggancia automaticamente alla rete UMTS, dandoci la possibilità di trasferire audio e video ed anche di navigare in Internet mediante il nostro PC portatile al quale si collega via Bluetooth.

Affollamento dello spettro

Uno dei problemi più evidenti che sorge con il diffondersi dei dispositivi wireless è quello delle interferenze: infatti si ha un progressivo riempimento delle bande disponibili e con questo bisognerà fare i conti nel futuro prossimo. Basti pensare che Bluetooth e Wi-Fi attualmente condividono la medesima banda ISM a 2,4GHz. In pratica, quello che succede facendo operare contemporaneamente nella stessa area apparati Bluetooth e Wi-Fi, è una interferenza che finisce con abbassare le prestazioni in termini di velocità di trasferimento dei dispositivi. Sono, comunque, già stati pensati dei metodi che consentano di mitigare gli effetti delle interferenze, introducendo nei sistemi Bluetooth un Frequency Hopping intelligente, in cui prima di occupare un canale ci si accerta che sia effettivamente libero, cercando in questo modo di non sovrapporsi ad un eventuale canale Wi-Fi. Nel prossimo futuro verranno molto probabilmente rese disponibili nuove bande di frequenza per queste applicazioni, che consentiranno un ulteriore sviluppo.

Nuove tecnologie

La ricerca segue anche altre direzioni ed una delle tecnologie più promettenti per i prossimi anni è l'**Ultra Wide Band** (UWB, www.uwb.org) che è rivoluzionaria rispetto ai sistemi di comunicazione che si basano sulla modulazione di onde sinusoidali. Infatti nell'UWB non si ha più un'onda continua, ma delle sequenze di impulsi di brevissima durata che hanno corrispondentemente un amplissimo spettro di frequenza che supera i 10GHz.

I promotori dell'UWB sostengono che una tecnica di trasmissione di questo tipo possa raggiungere agevolmente velocità di trasferimento di centinaia di megabit/s, ma il problema da risolvere è assicurarsi che questi impulsi ad ampio spettro non vadano a disturbare le comunicazioni tradizionali che lavorano nelle diverse zone dello spettro, si pensi in particolare alla strumentazione degli aerei, al sistema GPS ed ai radar.

CONCLUSIONE

Abbiamo visto quali sono le principali tecnologie wireless digitali e quali sono le loro principali applicazioni di oggi.
Sicuramente questo è uno dei settori in cui i produttori di microelettronica sono più attivi e quindi le novità sono all'ordine del giorno con dispositivi sempre più piccoli e preformanti.

CMOS

CCD B/N

CMOS

CCD
a colori

EXPLORER: I ROBOT ESPLORATORI

di Raffaello Bonghi
Vegekou@fastwebnet.it

Nell'articolo precedente abbiamo introdotto i robottini da sumo. Oggi ci occuperemo della categoria dei robot Explorer: i più diffusi nelle scuole ed in particolar modo negli istituti tecnici.

DI COSA SI TRATTA?

Torniamo indietro nel tempo e trasferiamoci in Grecia, quasi 2000 anni prima della nascita di Cristo. In quel tempo, all'epoca del Minotauro, i cretesi combattevano contro i greci. Fu allora che Minosse, re dell'isola di Creta, diede incarico a Dedalo di costruire un labirinto per rinchiudervi il Minotauro; il labirinto avrebbe dovuto essere talmente intricato da impedire a chiunque di uscirne. Dedalo, nella speranza di guadagnarsi la fiducia del sovrano, costruì quindi l'edificio noto alla storia come il Labirinto di Cnosso.
Vuole poi la leggenda che il Minotauro venisse rinchiuso nel labirinto e che, ogni anno, sette giovani e sette fanciulle di Atene, sconfitta dal re di Creta, gli venissero sacrificati per saziare la sua fame di carne umana.
Il sacrificio fu ripetuto due volte, ma la terza volta giunse a Creta Teseo, figlio di Etra e del sovrano di Atene Egeo, con l'intento di porre fine ai sacrifici. L'impresa era molto difficile non solo perchè avrebbe dovuto uccidere il Minotauro, ma anche perchè, una volta entrati nel Labirinto, era quasi impossibile uscirne.
Teseo, per poter raggiungere Creta, finse di essere uno dei fanciulli da sacrificare. A Creta il giovane si innamorò di Arianna, figlia di Minosse, che lo aiutò nell'impresa. Quando venne il suo turno egli entrò nel labirinto dipanando lungo la strada un rocchetto di filo, fornitogli da Arianna. Teseo giunse al cospetto del mostro, lo uccise e, riavvolgendo il filo, riuscì ad uscire dal labirinto. Finì così l'orrendo sacrificio che era stato

Figura 1: *Una Robofesta*

imposto da Minosse agli ateniesi e Teseo ed Arianna fuggirono da Creta e si recarono a Nasso.
Non appena sbarcati, Teseo dichiarò ad Arianna che aveva finto di amarla per salvarsi dalla prova del labirinto e abbandonò la fanciulla sulla spiaggia.

Lo spirito delle competizioni dei robot Explorer rispecchia la ricerca del Minotauro da parte di Teseo. Nelle gare il Minotauro è rappresentato da un serie di fonti di luce, suono e gas che vengono disposte lungo un labirinto insidioso e pieno di ostacoli. Il principale avversario dei robottini è il tempo a loro disposizione: solo tre minuti. Entro questo tempo gli Explorer devono individuare tutte le sorgenti e contemporaneamente devono saper evitare, possibilmente ricordandoli, gli ostacoli.
Scenderemo adesso nel dettaglio e vedremo come si costruisce, in linea generale, un robot Explorer. Vedremo inoltre come è organizzata una gara e quali ne sono le regole.

LE CATEGORIE

Esistono tre categorie di Explorer:

1 Explorer Junior Programmabili
2 Explorer Junior Analogici
3 Explorer Senior

Explorer Junior

Le due categorie Explorer Junior sono nate per spingere i ragazzi del 2° anno delle scuole superiori a cimentarsi nella costruzione di piccoli robottini esploratori. Il campo e le regole sono molto più semplici di quelli della categoria Senior, ciò non toglie che la difficoltà di costruzione e l'impegno richiesto siano quasi equivalenti.
Il campo di gara è di 2 metri per 2. Le misure dei robot non possono superare i 20 x 20 x 25 cm (sensori di contatto esclusi) ed i limiti vengono fatti rispettare in modo rigoroso.
I nostri robottini devono cercare sei fonti di luce disposte casualmente all'interno del labirinto. Non appena un robot trova una fonte di luce, il direttore di gara la spegne, per consentire al robot di riprendere la ricerca delle fonti di luci rimanenti.
Ma cosa hanno di differente le categorie degli Explorer Junior Programmabili e degli Explorer Junior Analogici?
L'elettronica di controllo!
Nella prima categoria, vale a dire quella degli "Explorer Junior Programmabili", i movimenti del robot e la gestione dei sensori viene effettuata da un microcontrollore; tra i più comunemente utilizzati ci sono i controller integrati della Microchip, grazie alla semplicità dello schema elettrico.
Mentre, per quanto riguarda gli "Explorer Junior Analogici", i sensori ed i motori vengono controllati in modo puramente analogico, quindi con relè, transistor e diodi e senza il supporto di un programma.

Figura 2: *Campo Explorer junior*

Explorer Senior

La categoria degli Explorer Senior è quella più complessa ed impegnativa, ma più interessante.
Questi robot devono percorrere un labirinto chiuso, quindi senza vie di uscita, di 3 metri per 2. All'interno del labirinto essi devono cercare e trovare tre fonti di luce, tre fonti di suono e tre fonti di gas (il gas in questione non è altro che alcool etilico). Ogni volta che trovano una di queste sorgenti, i robot devono illuminare i propri LED e fermarsi per tre secondi, in modo tale da permettere agli arbitri ed ai giudici di

convalidare il punto.

I colori utilizzati sono standard e sono indicati dal regolamento:

Colore	Sorgente	Punteggio
Rosso	Gas	1
Verde	Luce	2
Giallo	Suono	3

Le misure di questi robot devono rientrare entro un cubo di 30 cm di lato. La principale difficoltà che i costruttori devono affrontare è la scrittura di un programma di controllo che sia in grado, gestendo un opportuno numero di sensori, di individuare le sorgenti, accorgersi delle pareti e degli ostacoli e non perdersi nei meandri del labirinto.
Ma non è tutto! Le cose sono ancora più complicate. Infatti i robot devono percorrere il labirinto mentre tutte le fonti di luce, di suono e di gas sono sempre attive. Il robot deve quindi sapersi allontanare da una sorgente per ricercare una nuova fonte e deve evitare di individuare la stessa fonte più di una volta. Chi scrive il software deve quindi realizzare un buon sistema di mappatura del labirinto e delle sorgenti.

Nella figura 3 c'è un esempio di campo Explorer che, come si può vedere, è ricco di vicoli ciechi, di ostacoli e di sorgenti molto vicine tra loro. I semicerchi o i cerchi delimitano lo spazio in cui il robot si deve fermare per guadagnare il punto relativo alla sorgente.

Inoltre, per ogni errore, i robot subiscono una penalità consistente:

Ostacolo	Punteggio
Ostacolo Generico	-1
Sorgente o Ostacolo contenente la sorgente	-2

DI COSA C'È BISOGNO PER POTER COSTRUIRE UN ROBOT EXPLORER?

Innanzitutto sono necessari sensori per la rilevazione della presenza di pareti od ostacoli: i più utilizzati sono i sensori infrarossi per le lunghe distanze ed i sensori di contatto, chiamati "baffi", che permettono di accorgersi dell'imminente urto con un ostacolo e di cambiare percorso.
I sensori più utilizzati per trovare le fonti di luce sono i fotoresistori, particolari componenti che cambiano la loro resistenza quando

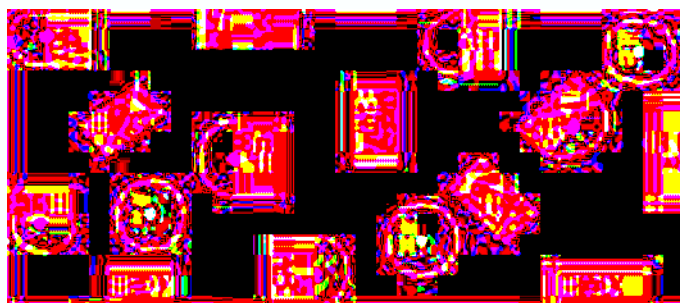



Figura 3: *Campo Explorer senior*

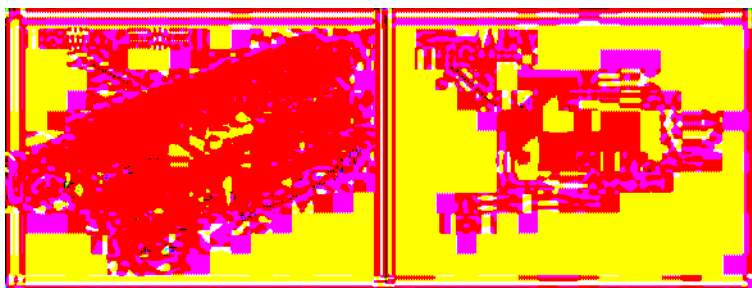



Figura 4: *Sensori di contatto a infrarossi*

Figura 5: *Fotoresistori e fotodiodi a confronto*

sono irradiati da una luce, oppure i fotodiodi, che conducono quando vengono colpiti dalla luce.
Per l'individuazione dei suoni vengono utilizzati microfoni tarati per ricevere soltanto onde di 4 KHz, per non confondere il robot con il suono delle voci del pubblico durante le competizioni.

Figura 6: *Microfono*

Infine, per poter trovare all'interno del labirinto le fonti di alcool etilico, vengono utilizzati molto spesso sensori del tipo di quelli utilizzati in cucina per gli allarmi contro le fughe di gas.
Nessuna delle categorie Explorer è soggetta a limiti di peso: questo permette di montare motori più potenti e batterie più pesanti.
Le batterie più usate sono quelle al piombo: grazie alla loro lunga durata ed al loro notevole amperaggio, infatti, riescono a mantenere attivi tutti i sensori per tutto il periodo di gara.

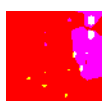
Figura 7: *Sensore del gas*

CONCLUDENDO

La categoria dei robot Explorer è la più istruttiva ed interessante, ed infatti è la più adottata negli Istituti Tecnici; consente infatti di mettere in pratica ciò che è stato insegnato durante l'anno scolastico.
Chi vuole avere maggiori informazioni sulle diverse categorie Explorer può chiedere nel nuovo forum di www.minisumo.net. All'interno del forum professori e partecipanti alle competizioni vi potranno dare consigli per costruire il miglior robot possibile e magari per partecipare e vincere nelle gare di categoria.

BIBLIOGRAFIA

www.minisumo.net il portale italiano delle gare di robotica in Italia

ANCHIP III ROBOT EXPLORER SENIOR	
Nome:	AndChip III
Costruttore	Andrea Massimi (andmassi@aliceposta.it)
Microntrollore	PIC16F877
Motori	2 motori ridotti controllati da un ponte H (L293D)
Sensori di Luce	5 Fotodiodi
Sensori di Suono	2 Mic. Amplificati e controllati da un ToneDecoder (NE567
Sensori di Gas	ST-11 Amplificato (LM358)
Altezza	195 mm
Larghezza	190 mm
Profondità	175 mm
Sensori Urto	Sharp GP2D120, baffi
Metodo gestione sensori	• Luce: Analogico (0V=buio, 5V=max luce) • Suono: Digitale (0= suono a 4 khz, 1= altri suoni) • Gas: Analogico (Val. crescente quando vi è GAS) • Urto: Analogici (0V= nessun ostacolo, 3.5V= ostacolo vicino)

FEBOT quarta parte

IL FIRMWARE DELLA SCHEDA MADRE

di ***Maurizio Del Corso*** *(m.delcorso@farelettronica.com)*
e ***Tiziano Galizia*** *(t.galizia@farelettronica.com)*

Concludiamo la trattazione della scheda madre del Febot analizzando le routines di invio e ricezione dati sul bus seriale e la routine di gestione del LED ERROR. Anche in questo articolo non saranno riportate intere sezioni di codice, ma verrà fatto largo uso dei diagrammi di flusso per chiarire le tecniche adottate. In questa versione di firmware sono state implementate le funzioni minime, ma contiamo sulla vostra collaborazione per rilasciare versioni sempre più complete.

DOVE ERAVAMO RIMASTI

Il mese scorso abbiamo visto come è strutturato il firmware della scheda madre, quindi analizzato i vari file var.inc, library.inc e variabili.inc. Per quanto riguarda le routines, sono state analizzate quelle relative alle interruzioni e quella per la lettura e conversione in digitale della tensione di alimentazione.
Dedichiamo ampio spazio questo mese alle routines di invio e ricezione dati dal bus seriale.

Svincolarsi dal clock

Il protocollo di comunicazione scelto per lo scambio dati tra i

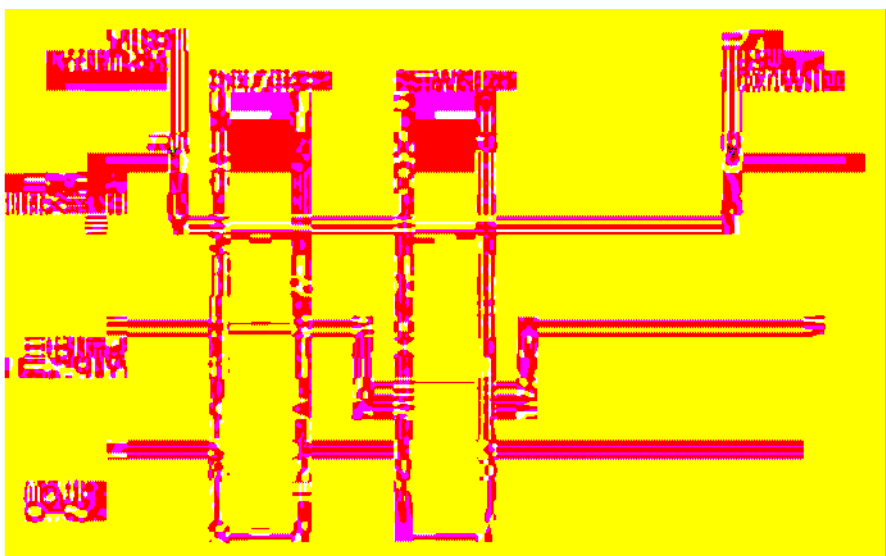



Figura 1: *Trasmissione di un bit "1" e un bit "0"*

vari moduli del Febot, è stato pensato in modo da essere indipendente dalla velocità di elaborazione dati del controllore a bordo di ogni modulo. Questo permette la corretta comunicazione anche tra due moduli che utilizzando due microcontrollori diversi e operanti a frequenze diverse. Prima di analizzare le operazioni di invio e ricezione dati dal punto di vista firmware, vale la pena chiarire come la comunicazione sia possibile anche con diverse frequenze di clock.
In figura 1 è riportata la temporizzazione sul bus seriale nell'ipotesi in cui si stia trasmettendo un livello logico alto e, successivamente uno basso.

Innanzitutto prima dell'inizio della trasmissione è opportuno verificare la disponibilità del bus analizzando il bit BUSY_S. Solo se questo è a livello logico alto sarà possibile iniziare la trasmissione.
All'inizio della trasmissione, si pone a livello logico basso la linea BUSY_S dopodichè si procede con l'invio secondo il seguente algoritmo:

1 Si pone CLK a livello alto.
2 Si prepara il bit da trasmettere impostando opportunamente il livello logico della linea S_DATA.
3 Si pone CLK a livello basso.
4 Si attende un intervallo di tempo opportuno.
5 Si riporta CLK a livello alto.

In questo modo, il modulo ricevente deve analizzare la linea CLK e leggere il dato sulla linea S_DATA solo quando CLK è a livello basso. La velocità della comunicazione è dunque stabilita dal trasmettitore attraverso il segnale CLK.

Sfruttando questo accorgimento, la comunicazione dati può avvenire senza problemi tra due PIC pilotati, ad esempio, con quarzi a 4MHz e 20MHz.

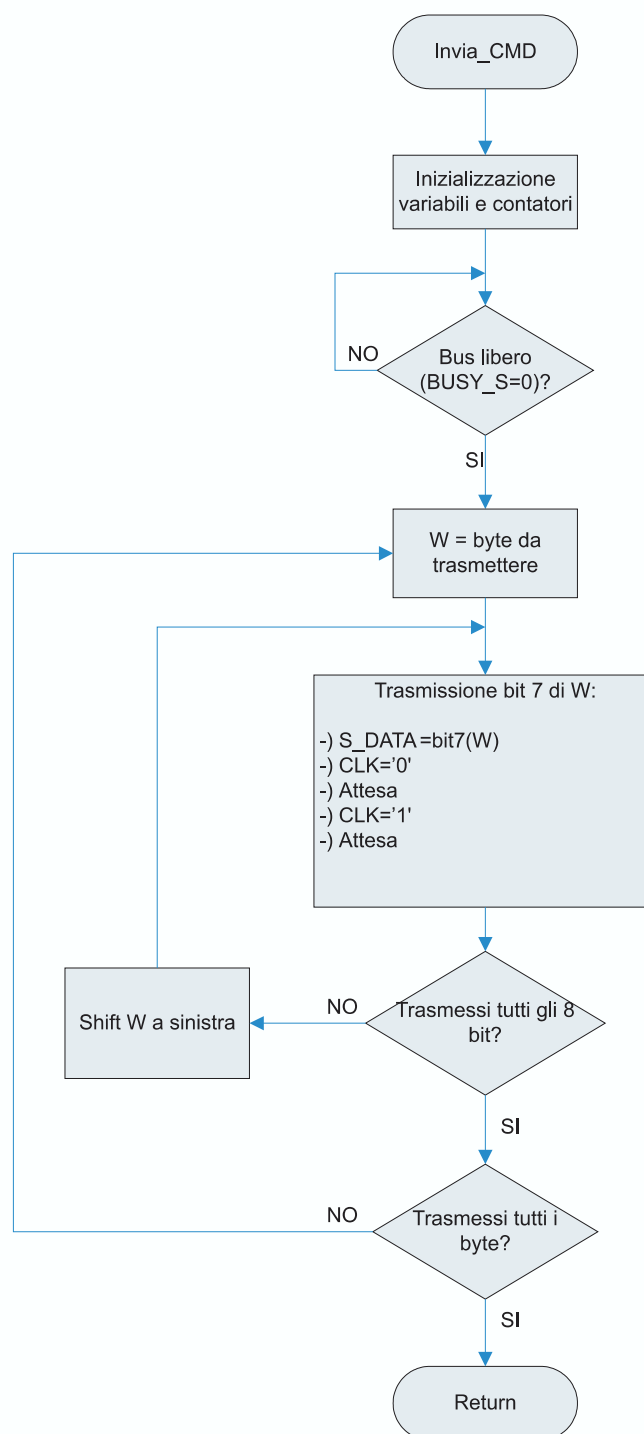
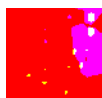



Figura 2: *La routine "Invia_CMD" per la trasmissione di un dato*

È importante che la linea S_DATA sia stabile al valore da trasmettere, per tutto l'intervallo di tempo in cui il segnale CLK rimane a livello basso.

L'invio dei dati

La routine *Invia_CMD* provvede a pilotare il bus di trasmissione secondo quanto specificato in precedenza. Il diagramma di flusso per questa routine è riportato nella figura 2. Poiché la trasmissione di un pacchetto dati, comporta in realtà la trasmissione di 19 byte, è necessario mantenere il conteggio dei bit via via trasmessi.
La tecnica utilizzata in questa routine è quella di trasmettere in continuazione il bit 7 del registro accumulatore W (il bit più significativo) quindi il contenuto di W deve essere dapprima caricato con il byte da trasmettere, quindi una volta trasmesso il primo bit, se ne deve traslare il contenuto verso sinistra affinché il bit da trasmettere divenga il più significativo di W.

Il segnale CLK viene mantenuto a livello basso per 1 millisecondo, in modo da consentire, a chi riceve, di effettuare la ricezione del dato in maniera corretta.
Ovviamente questo tempo determina la velocità di trasmissione e non può essere troppo basso per non compromettere la ricezione. Il tempo di attesa viene gestito dalla routine *Dlay1* che è stata calibrata per un quarzo da 4MHz. Il 16F876 esegue una istruzione in 4 cicli ci clock per cui la routine *Dlay1* è stata fatta in modo da eseguire precisamente 1000 istruzioni impiegando quindi un tempo pari ad 1 millisecondo.
È evidente che se nei vostri progetti utilizzate un quarzo di valore diverso, tutte le routine di generazione dei ritardi andranno riviste e modificate.
Lo scorrimento verso sinistra del registro W, viene effettuato con l'istruzione ***rlf*** (Rotate Left f through Carry). Nello spostamento di tutti i bit verso sinistra il bit più significativo viene memorizzato nel bit Carry (C) del registro STATUS ed il precedente valore del bit carry viene caricato come bit meno significativo (vedi figura 3). In questo modo dopo 8 operazioni di shift a sinistra su W (o su un generico registro *f*), ne viene ripristinato il contenuto.

Ovviamente esiste anche l'operazione inversa (rrf) che provoca lo scorrimento del contenuto del registro verso destra caricando con il valore del bit Carry, il bit più significativo del registro (figura 4). Lo operazioni di shift a destra e shift a sinistra sono molto utili in quanto consentono di eseguire moltiplicazioni o divisioni per due sul contenuto di un registro (dividere per due in binario corrisponde infatti a traslare tutto di un bit verso destra ed inserire uno zero in testa, viceversa moltiplicare per due un numero binario è equivalente a spostare tutto il contenuto verso sinistra di un bit ed inserire uno zero in coda).

Per le due operazioni, effettuate su un registro generico f, è possibile scegliere dove memorizzare il risultato. Le alternative sono il registro stesso oppure il registro accumulatore W.

Ricezione dei dati

La ricezione dei dati è gestita dalla routine *Ricevi_CMD* il cui diagramma di flusso è riportato in figura 5.
Con questa routine viene analizzata la linea CLK perché, come già detto, quando questa va a livello basso significa che il dato presente sul filo S_DATA è pronto per essere letto. Dunque se CLK vale 0, viene letta ciclicamente la linea

Figura 3: *L'operazione RLF sul registro generico f*

Figura 4: *L'operazione RRF sul registro generico f*

S_DATA fintanto che CLK non torna ad 1. Se S_DATA vale 0 viene caricato il valore "00000000" in W altrimenti viene caricato il valore "00000001". Quando CLK è tornato ad 1, viene elaborato il dato letto aggiornando l'opportuna variabile a seconda del numero di bit ricevuti.

Alla fine viene calcolato il CRC in locale e se questo coincide con quello ricevuto nel pacchetto, la procedura termina lasciando il valore 0 in W. Nel caso in cui il CRC non coincide, in W viene lasciato il valore 1.

La ricezione avviene un bit per volta, quindi abbiamo la necessità di aggiornare le varie variabili via via che si ricevono i vari bit. La tecnica usata per l'aggiornamento di un singolo bit di una variabile di 8 bit, è quella della mascheratura: se il bit letto è "1" viene fatto un OR tra la variabile da aggiornare e il valore "00000001" (01H) altrimenti viene fatta una operazione di AND con il valore

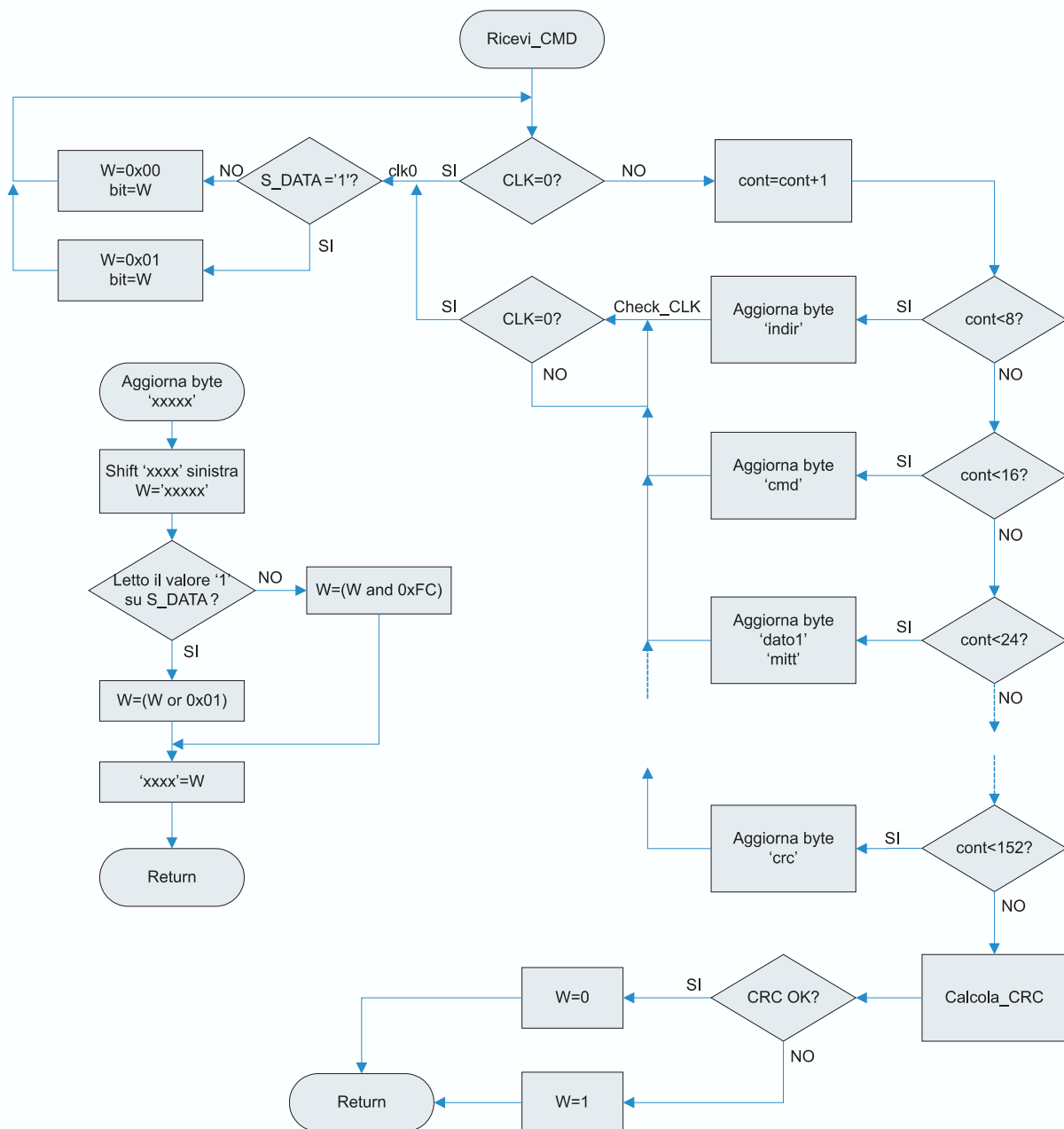



Figura 5: *La routine "Ricevi_CMD" per la ricezione di un dato*

“11111110” (FCH). È noto che l’operazione di OR dà come risultato 1 se almeno uno dei due operandi vale 1. Fare un OR tra una variabile ed il valore 00000001 significa dunque mettere ad 1 il bit meno significativo della variabile senza modificare gli altri bit. La variabile viene quindi fatta scorrere di un bit verso sinistra per essere pronta alla ricezione del bit successivo. L’operazione di AND fra due operandi dà come risultato 0 se almeno uno dei due operandi vale 0.
L’operazione di AND tra la variabile ed il valore “11111110” provocherà quindi l’azzeramento del bit meno significativo della variabile lasciando gli altri inalterati.

Figura 6: *La routine “Errore” per la gestione del LED ERROR*

Calcolo del CRC

Il campo CRC del pacchetto dati è necessario per stabilire l’integrità del pacchetto. Nella versione 1.0 del firmware questa funzione non è implementata pur essendo prevista la routine che in realtà fornisce sempre un risultato positivo indipendentemente dal valore del campo CRC. Nella versione successiva verrà implementata la tecnica del checksum per la verifica del pacchetto. Vediamo in dettaglio di cosa si tratta. Il dispositivo che trasmette il pacchetto dati, costruisce il pacchetto da trasmettere inserendovi i vari byte. L’ultimo byte, il campo CRC appunto, viene calcolato alla fine utilizzando i valori precedentemente caricati nel pacchetto. In particolare vengono contati i bit ad 1 ed il risultato viene memorizzato nel campo CRC. Chi riceve il pacchetto dovrà fare altrettanto: verranno contati i bit ad 1 di tutti i byte del pacchetto ricevuto (escluso il byte CRC) ed il risultato verrà confrontato con il campo CRC del pacchetto ricevuto. Se i due risultato sono diversi, significa che c’è stato un errore nella ricezione dei dati. Ovviamente se i due risultati coincidono, non è detto che il pacchetto sia stato ricevuto correttamente, ma la probabilità che ci sia stato un errore è comunque molto bassa. Questa è la stessa tecnica usata nei protocolli di trasmissione come il TCP.

Gestione del LED ERROR

Qualora il controllo CRC non abbia dato risultato positivo, viene richiamata la routine *Errore* che farà lampeggiare per tre volte a distanza di mezzo secondo, il LED ERROR della scheda madre. Questa routine è molto semplice e sfrutta un ciclo di tre iterazioni di 500ms in ognuna delle quali viene variato lo stato dell’uscita ERR (a cui è connesso il led). In ogni caso, prima dell’uscita dalla routine, il LED viene comunque spento. In figura 6 il diagramma di flusso della routine *Errore*.

CONCLUDIAMO

Con questo articolo abbiamo esaurito le linee guida relative al firmware della scheda madre.
Potete scaricare i sorgenti dal sito di Fare Elettronica (www.farelettronica.com) e vi invitiamo ad apportarvi miglioramenti al fine di incrementare le potenzialità e l’ottimizzazione del sistema. Se avete proposte o avete rielaborato le routine proposte, non esitate a contattare il servizio Mailbox (mailbox@farelettronica.com) e ricordate che, per l’ottica di questo progetto, il vostro contributo è fondamentale!

INTRODUZIONE ALLA ROBOTICA

quinta parte

LA MATEMATICA, L'ANTROPOMORFISMO E LA PROGRAMMAZIONE

di Massimilinao Bracci
m.bracci@farelettronica.com

Con questa ultima puntata si conclude "Introduzione alla Robotica". Spero di non avervi annoiato ma di aver trasmesso un buon movente affinché possiate approfondire il settore della robotica in tutte le sue svariate argomentazioni.

LA MATEMATICA DEI ROBOT

La parte meccanica dei robot, come già detto in precedenza, è formata da un certo numero di elementi rigidi indicati spesso con il nome di origine americano links, connessi tra loro tramite giunti detti anche joints. Questi collegamenti sono usualmente realizzati (specialmente in campo industriale) in modo tale che ogni links abbia normalmente, rispetto al resto del sistema che lo precede, un solo grado di libertà.
Se i giunti risultano essere in numero di N e ognuno fornisce un grado di libertà al braccio che segue (rispetto a quello che lo precede), i links sono in una quantità di N+1 (con il primo fisso solidale alla base) e i successivi N sono in movimento.

Lo studio cinematico degli arti, dal punto di vista robotico, inizia con considerazioni di carattere geometrico. Osservata la struttura dell'arto, quindi riconosciuti e modellati i giunti è possibile sistemare delle terne cartesiane solidali con i tratti di

Figura 1: *Articolazione con giunti*

gamba (figura 2) o di un braccio (figura 3) che, a questo punto, si considerano come corpi rigidi in modo da fissare uno zero relativo e definire la posizione dei vari punti.

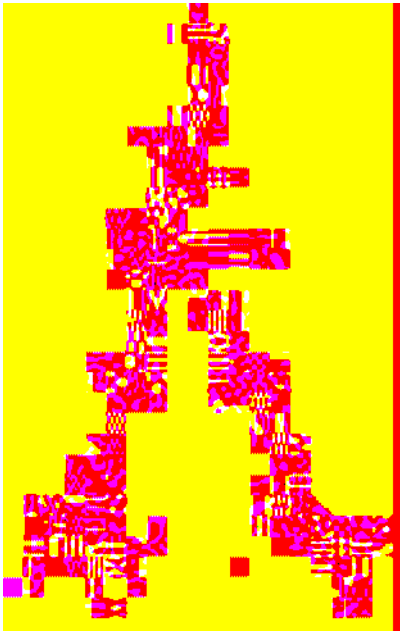
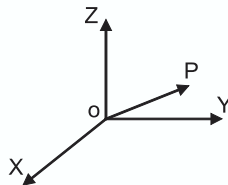



Figura 2: Schema statico del corpo e gambe di un robot antropomorfo

Per definire la posizione dei vari punti di spostamento e per individuare le rette delle direzioni nello spazio, rispetto ad un determinato sistema di riferimento, si utilizza la rappresentazione mediante i vettori e per rappresentare le rotazioni e gli spostamenti si utilizzano le matrici.

Consideriamo un punto **P** nello spazio di coordinate **a,b,c,** spazio definito dalla terna di assi cartesiani x,y,z dove la direzione è stabilita dai versori **i, j, k,** è possibile individuare un vettore posizione **V =a*i+b*j+c*k**. Tale vettore risulta essere il segmento che unisce l'origine O degli assi con il punto **P** (con direzione da O verso P) rappresentato mediante coordinate omogenee con una matrice colonna (vettore colonna):

$$V = \begin{vmatrix} f \\ l \\ m \\ w \end{vmatrix}$$

Figura 3: Terna cartesiana di riferimento su mano

Dove f=a*w, l=b*w, m=c*w e w= fattore di scala necessario per poter effettuare il prodotto della matrice colonna con le matrici di trasformazione.

La trasformazione di un punto o di un vettore posizione individuato nello spazio da una terna cartesiana può essere espresso mediante l'applicazione di una matrice di trasformazione: dato ad esempio un vettore **T = d*i+e*j+f*k** la sua traslazione è effettuata con:

$$H = trasl\ (d,\ e,\ f,) \begin{vmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & e \\ 0 & 0 & 1 & f \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Le matrici di trasformazione che esprimono le rotazioni di un angolo di ampiezza θ attorno ad ognuno degli assi cartesiani x, y, z della terna di riferimento assumono le seguenti forme:

$$Rot.(x,\ \theta) = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\mathrm{sen}\theta & 0 \\ 0 & \mathrm{sen}\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$Rot.(y,\ \theta) = \begin{vmatrix} \cos\theta & 0 & \mathrm{sen}\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\mathrm{sen}\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$Rot.(z,\ \theta) = \begin{vmatrix} \cos\theta & -\mathrm{sen}\theta & 0 & 0 \\ \mathrm{sen}\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Si prenda di riferimento un vettore posizione V, definito nello spazio da un sistema di riferimento cartesiano x,y,z, con i versori i,j,k:

V = 2*i+3*j+1*k

Si pensi di eseguire una rotazione del vettore V attorno all'asse z (verticale) con un angolo di 90°.
Utilizzando la matrice H1: matrice Rot (z, θ):

$$U = H1 * V = \begin{vmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 3 \\ 2 \\ 1 \\ 1 \end{vmatrix}$$

Ottenendo il vettore spostamento:

$$U = \begin{vmatrix} -3 \\ 2 \\ 1 \\ 1 \end{vmatrix}$$

Se si pensa poi di ruotare il vettore **U** di un angolo do 90° attorno all'asse y ottenendo un vettore **W**.
Usiamo la matrice rotazione Rot.(y, θ):

$$W = H2 * U = \begin{vmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} -3 \\ 2 \\ 1 \\ 1 \end{vmatrix}$$

$$W = \begin{vmatrix} 1 \\ 2 \\ 3 \\ 1 \end{vmatrix}$$

Non mi addentrerò ulteriormente nel fantastico (o dannato) mondo dell'algebra matriciale per evitare che questa possa divenire una pesante lezione. Lascio lo stimolo a voi appassionati così che possiate, con non poca fatica, approfondire l'argomento in modo individuale.

La trasformazione tra un sistema di coordinate ed il suo precedente permette di risalire, nota la posizione del sistema di riferimento di base, all'esatta localizzazione di tutti i punti della struttura robotica. Le relazioni che legano la rototraslazione di un sistema di riferimento rispetto ad un altro si basano, normalmente, sul metodo matriciale di Denavit & Hartenberg. Le matrici di trasformazione che si ottengono con questo metodo vengono indicate con il nome matrici ***D-H***. e tale rappresentazione consiste appunto nel realizzare una matrice di trasformazione omogenea $^{i-1}[D\text{-}H]_i$ appartenente R^{4x4} che rappresenta il sistema di coordinate del *link i-esimo* rispetto al *link* precedente (i-1)-esimo.
Per poter applicare questo tipo di soluzione occorre che tutte le articolazioni siano formate da superfici che possono scorrere una sull'altra senza abbandonare il contatto tra loro. Questi tipi di giunti vengono detti lower-pair e sono:

- Di rotazione semplice: prendendo un unico movimenti attorno ad un asse.
- *Prismatici*: con traslazione semplice.
- *Cilindrici*: con due spostamenti, uno di traslazione ed uno di rotazione tra loro indipendenti.
- *A vite*: ancora con due spostamenti ma con la traslazione dipendente dall'entità della rotazione.
- *Planari*: con traslazione comunque orientata su di un piano.
- *A sfera*: con rotazione attorno ad un punto.

Oltre quindi alle condizioni costruttive delle articolazioni, per applicare il metodo di D-H occorre anche assegnare ad ogni giunto del braccio un sistema di riferimento mobile con il giunto stesso.Quindi la frame del giunto si sposta solidale all'articolazione a cui è associata, per effetto del movimento degli elementi che precedono il giunto nella catena cinematica, ma è fissa rispetto al tipo di spostamento effettuato dall'elemento successivo per effetto del moto del giunto in oggetto. In questo modo è possibile determinare la posizione dell'elemento rigido successivo rispetto al sistema di riferimento associato al giunto stesso (figura 1).

Il passo successivo veramente arduo da affrontare, una volta risolto il problema della cinematica è quello della modellazione dinamica.
La cinematica del robot come già detto riesce a descrive la geometria ed il movimento del manipolatore e di ogni altro giunto (a prescindere dalle cause che lo hanno generato) nei due modi seguenti:

- *Diretto*: descrivendo la posizione e l'orientamento dell'organo interessato come una funzione (non lineare) delle variabili del giunto.
- *Differenziale*: descrivendo la velocità lineare ed angolare del l'organo interessato come funzione della velocità del giunto.

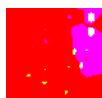
La dinamica del robot descrive la relazione tra le cause (coppie

e forze) ed il loro effetto sul movimento del giunto e anch'essa può essere sviluppata in due modalità:

- *Diretta*: date per note le coppie e le forze si determina il movimento.
- *Inversa*: dato un certo movimento si calcolano le forze necessarie per eseguirlo.

Per ricavare il modello matematico-dinamico di sistemi meccanici articolati a più gradi di libertà esistono diversi approcci e tra i più importanti viene da ricordare quello di Eulero-Lagrange dove, mediante operatori matematici complessi, si va a descrivere la formulazione energetica dei vari corpi rigidi (giunti, bracci, eccetera) calcolando l'energia cinetica e quella potenziale dovuta ad una determinata azione.

L'ANTROPOMORFISMO

La tecnica di locomozione animale largamente più diffusa in natura è quella che prevede l'utilizzo di gambe, piedi, zampe e in generale di arti (due, quattro, sei, otto, mille) e nella fattispecie quella che risulta essere la più evoluta ne prevede esattamente due.
La necessità di realizzare un robot (domestico e non) di concezione antropomorfa viene classicamente illustrata con il problema delle scale. Sebbene sia possibile studiare particolari attrezzature che permettano al robot di superare scale e gradini queste normalmente risultano una grande limitazione nelle funzionalità della macchina e comunque non permettono di risolvere il problema degli spostamenti sui terreni sconnessi.
Fornendo al robot lo stesso tipo di locomozione di cui siamo dotati, possiamo evidentemente chiedergli di manovrare ed operare negli stessi ambienti nei quali noi operiamo.
La concezione di un robot mobile deve quindi tenere in considerazione il tipo di spazio

nel quale la macchina dovrà agire e se il nostro progetto vuole avere la massima applicabilità e flessibilità allora il robot dovrà necessariamente avere delle gambe.
Quante gambe?
La progettazione di un robot parte da una specifica sulle funzionalità che la macchina deve avere, ovvero dalla base dei compiti ai quali sarà poi destinato e la maggior parte dei compiti robotici oramai richiede un certo dinamismo per essere competitiva nei confronti di un equivalente abilità umana. Se la tecnica meccanica permette di sfruttare frequenze di lavoro molto spinte lo stesso non vale per la tecnologia del controllo elettronico: per seguire l'evoluzione dinamica di un sistema complesso come un robot mobile sono necessarie frequenze computazionali decisamente elevate.
È per questo che un soddisfacente compromesso può essere quello di realizzare robot dinamicamente stabili in modo intrinseco, così da poter concentrare le risorse di controllo in uno sforzo più specifico.
Il numero di gambe può essere grande a piacere, se è vero che all'aumentare del numero di gambe l'equilibrio dinamico è maggiormente garantito è anche vero che i gradi di ridondanza del sistema da gestire contemporaneamente annullano, di fatto, l'utilità di averne così tante.
Le scelte possibili si riducono quindi a robot bipedi, a tre zampe, quadrupedi, esapodi ed ottapodi; ma di questi ultimi non ne esistono molti.
I robot bipedi hanno ancora grandi problemi legati alla dinamica del camminamento che implica molto di più del muovere i piedi in modo da metterli uno davanti all'altro e quindi il loro campo di studio è ancora nella fase di ricerca di laboratorio.
I robot a quattro zampe permettono la stabilità statica della macchina, ma non l'uniformità del movimento per via del fatto che il margine di stabilità è normalmente piuttosto ristretto.
I robot esapodi garantiscono invece più facilmente sia la stabilità statica (in ogni momento almeno tre appoggi sono garantiti) che quella dinamica, maggiore flessibilità nel passo di camminamento, ma sono sicuramente più onerosi sia in termini di costo che di controllo a livello di problemi di coordinate.
I robot ottapodi rivestono attualmente una categoria particolare per applicazioni dove sono richieste doti di mobilità e di direzionalità, hanno inoltre il grande pregio di poter manovrare normalmente in spazi ristretti.

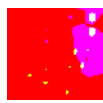
Vista la natura tridimensionale di ognuno dei *giunti* ci si può rendere conto come una semplificazione è strettamente necessaria, pena l'aumento della difficoltà del controllo dei gradi di libertà che si moltiplica con il numero delle gambe e del tipo di interazione che devono avere tra loro.
Fissiamo l'attenzione su di un arto in particolare.
I gradi di libertà dell'anca, ad esempio, che permettono un così ricco set di movimenti come quelli fisiologici vengono ridotti sulla base del tipo di compito e quindi del tipo di implementazione meccanica utilizzata per quel giunto, per il quale il robot è progettato. Un giunto che permetta la comple-

Figura 4: *Tipologie di articolazioni*

ta mobilità dell'arto può venire realizzato a patto di modificare però completamente quelle che sono le caratteristiche dell'ispirazione biologica dalle quali si vuole trarre spunto. I fattori stessi che mantengono a contatto e stabilizzano i capi articolari sono pressoché impossibili da replicare artificialmente: i legamenti e i fasci muscolari lavorano secondo una logica difficile da ricostruire con le tecniche ed i materiali che abbiamo a disposizione a livello costruttivo.

Dal punto di vista della semplice cinematica (figura 4) queste articolazioni possono dividersi in: articolazioni a superfici piane (artrodia), a superfici sferiche (enartrosi), a superfici ellissoidali (condilartrosi), a superficie curvilinea convessa e concava (articolazioni a sella) ed infine a superficie di contatto cilindrica (ginglimi laterali ed angolare).

Il robot WABIAN (WAseda BIpedal humANoid) dell'università giapponese di Waseda è una macchina alta 1.66m, pesa 107kg e prevede due gambe antropomorfe per un totale di 43 gradi di libertà. È capace di trasportare carichi fino a 2kg ad una velocità di 20cm/s. È movimentato utilizzando 15 motori a corrente alternata e ben 28 a corrente continua. Il consumo di questi attuatori non è stato dichiarato dai costruttori, ma è facile immaginare come solo la parte di potenza comporti richieste energetiche non indifferenti.	
Centaurob 1 della Paderborn University è un robot capace di trasportare fino a 50kg a fronte di un peso proprio di 90kg. È altro 1.7m e consuma 200W, è capace di raggiungere una velocità massima di 1cm/s.	
Il robot HITBWR-III (Harbin Inst. of Tech. Biped Walking Robot, cinese) segue una certa ispirazione biologica, modellando le articolazioni come giunti rotazionali accoppiati tra loro. Si muove ad una velocità di 20cm/s e consuma 400W. È alto 1m, pesa 40kg ed è capace di trasportare fino a 5kg.	
Altri tipi di macchine prendono invece ispirazione da altra natura biologica, per cui le articolazioni vengono modellate diversamente. Sono un esempio lo Spring Flamingo del Leg Laboratory dell'MIT ed il Meltran II, giapponese.	
Particolare cenno meritano poi alcuni robot la cui ispirazione biologica dall'uomo vuole essere particolarmente spinta. Esempi ne sono lo Shadow biped, del gruppo Shadow Robot Project, e l'Honda Human Robot, della Honda Motor Company.	

Tabella 1: *Alcune realizzazioni di Robot antropomorfi*

Per questo, in ambito robotica, si limita la funzionalità di ogni giunto sulla base del compito che l'arto deve svolgere, che il più delle volte è quello del puro camminamento piano
In tabella 1 potete notare alcune realizzazione di robot antropomorfi.

PROGRAMMAZIONE DI UN ROBOT

Programmare un robot vuol dire (specialmente in ambito industriale) dare ad esso tutte le istruzioni necessarie affinché compia una certa azione, magari mantenendo un certo orientamento, di aprire o chiudere una pinza, di attendere un segnale di consenso e questo può essere effettuato in modo tale da seguire una traiettoria ben precisa oppure una di tipo casuale.
Esistono diversi metodi adatti a programmare un robot, in genere si usa distinguere i metodi di programmazione in due tipi fondamentali; quelli di *istruzione manuale* e quelli *programmati con apposito linguaggio.*

L'istruzione manuale, detta anche programmazione per autoapprendimento, consiste nel portare l'arto interessato (usualmente la mano) del robot nei vari punti che dovranno essere raggiunti, nel far memorizzare al sistema di controllo tutte le posizioni desiderate e la velocità che dovrà essere mantenuta.
Il primo metodo di programmazione è stato quello *Teach-in* mediante tastiera: programmazione per autoapprendimanto. Si fa eseguire al robot lo spostamento desiderato memorizzandone gli spostamenti e la velocità.

Dallo schema si vede come durante la fase di apprendimento la pulsantiera in posizione 1 comanda l'attuatore interessato. Raggiunto la posizione desiderata il trasduttore di posizione, mediante il pulsante di memorizzazione, viene collegato alla memoria cosi da registrare i dati relativi

AUTOAPPRENDIMENTO: METODO TEACH-IN
memoria
pulsante di memorizzazione
tastiera di programmazione
sistema di controllo
selettore
1
2
ROBOT
attuatore
D/A
Sensori
A/D

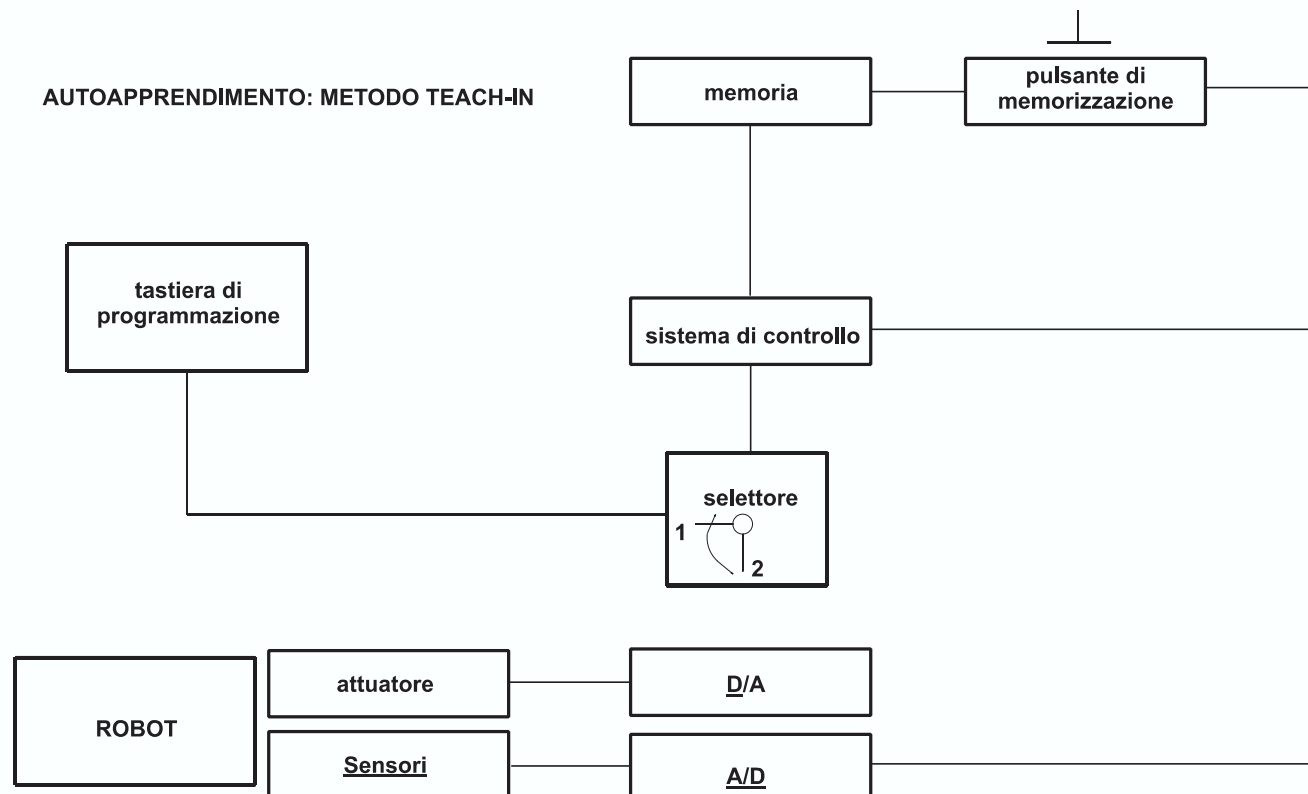



Figura 5: *Principio della logica di autoapprendimento Teach-in*

alla posizione. Commutando il selettore nella posizione 2 il robot può passare all'esecuzione cioè svolgerà i programmi utilizzando come segnali di riferimento le posizioni precedentemente memorizzate.
Con una tastiera di un pc può essere programmato un robot in grado di muovere un solo asse alla volta.
Nel caso in cui vi sia più assi da movimentare contemporaneamente tale metodo non è più idoneo. Può ad esempio essere impiegata la programmazione mediante joystick. In presenza di robot non cartesiani, tipo quelli articolati si può procedere ad una programmazione per autoapprendimento afferrando l'estremità del braccio del robot e guidandolo in modo che esso si posizioni ove desiderato. Una volta raggiunta la posizione i trasduttori di posizione dei singoli assi, mediante l'azionamento di un pulsante per la memorizzazione vengono collegati tramite scheda A/D con la memoria così che questa possa registrare i dati relativi alla posizione.

Il programma ottenuto, utilizzando uno dei molti *linguaggi di programmazione* ad alto livello, può essere inserito scrivendo le opportune istruzioni direttamente nella memoria dell'elaboratore di controllo. Le istruzioni variano a seconda del linguaggio di programmazione utilizzato e contengono le informazioni sulle posizioni da raggiungere, sul tipo di traiettoria da seguire, sulla velocità da mantenere durante lo spostamento… quindi per scrivere un programma di lavoro di un robot occorre utilizzare un linguaggio di programmazione che metta a disposizione una serie di istruzioni che gli permetta di individuare dei punti e angoli nello spazio e che siano in grado di gestire le informazioni provenienti dai sensori e dai trasduttori.
I metodi per raggiungere tali obiettivi sono usualmente due: quello di utilizzare un linguaggio di programmazione già esistente e adattarlo al nuovo sistema robotizzato, oppure quello di ideare un nuovo linguaggio.
La creazione di un linguaggio totalmente nuovo è, in ambito industriale, il metodo finora più seguito (soprattutto a causa della scarsa adattabilità dei linguaggi esistenti ad assumere il controllo di macchine tanto diverse tra loro, quali quelle attualmente disponibili in commercio).
Volendo scendere ancora nel dettaglio c'è da dire che i linguaggi di programmazione ad alto livello sono distinti tenendo conto del metodo che viene utilizzato dal sistema di controllo per comandare i movimenti delle articolazioni.
La classificazione usualmente utilizzata è quella che individua i diversi livelli:

- *Livello dei giunti:* si deve conoscere l'entità degli spostamenti eseguiti dagli attuatori che muovono appunto i giunti.
- *Livello della mano:* si deve conoscere la posizione ed i relativi spostamenti nello spazio (esempio: "portare la mano nel punto di coordinate X1, Y1, Z1").
- *Livello dell'oggetto:* il programma comunica al controllore quali sono gli oggetti interessati e quindi il robot deve possedere anche un sistema sensoriale in grado di distinguere gli oggetti.
- *Livello del compito:* oltre alle capacità precedenti il robot deve essere in grado di comprendere e interpretare le immagini.

In ambito sempre industriale, pur in presenza di svariati problemi di non facile soluzione, l'utilizzazione dei linguaggi di programmazione diviene indispensabile per coordinare tutti i movimenti dei robot gestiti da un cospicuo numero di sensori e trasduttori.

CONCLUSIONI

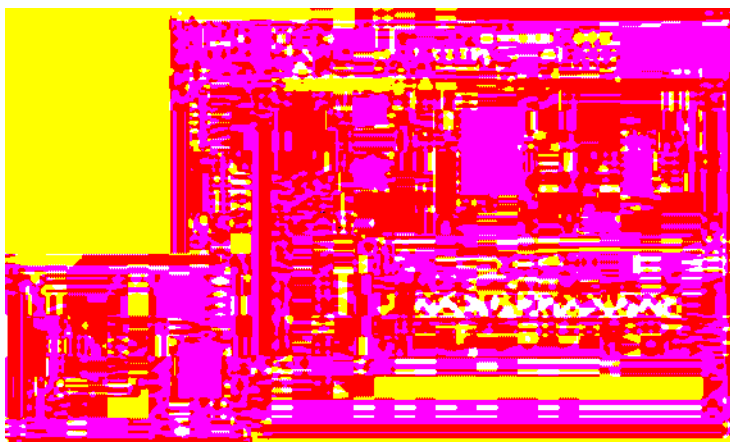
Siamo così giunti a termine di questa "Introduzione alla Robotica". Con questo articolo si conclude un percorso dove si sono analizzate le principali caratteristiche che identificano questo fantastico e vasto settore di tecnologia moderna.
L'argomento è molto complesso ed io ho cercato di dare un'idea e suscitare interesse in coloro che intendono avvicinarsi alla robotica con un minimo di approccio sistematico.
Ringrazio tutti i lettori che hanno inviato messaggi di apprezzamento per il lavoro svolto.

in vetrina

UN POTENTE AMBIENTE INTEGRATO PER LA PROGETTAZIONE ELETTRONICA

CadLogix costituisce un potente strumento per la progettazione elettronica. Dispone di funzioni avanzate per il disegno di simboli e schemi elettrici di alta qualità, per la gestione delle librerie di componenti e per la realizzazione dei modelli di simulazione. Include un simulatore analogico in standard SPICE e un simulatore digitale interattivo.

UN COMPLETO CAD 2D

L'editor grafico dispone di una vasta serie di funzioni che rendono CadLogix un potente CAD 2D per la realizzazione di disegni, simboli e schemi di alta qualità per l'elettronica. Si ha la possibilità di realizzare semplici schemi a foglio singolo oppure complessi progetti multifoglio (sia di tipo Flat che Gerarchico). Gestione di pagine multiple e di un numero illimitato di livelli di disegno. La barra del progetto e la finestra degli oggetti consentono di gestire facilmente anche progetti di grandi dimensioni e formati da numerosi oggetti. Include filtri per l'importazione di immagini e disegni e per l'esportazione nei formati grafici più diffusi consentendo di includere facilmente gli schemi realizzati in altri documenti per presentazioni e relazioni. Disegno di precisione grazie alla doppia griglia (grafica ed elettrica) e alle impostazioni di Snap. Include un controllo ERC (Electrical Rules Check) in linea che consente di evitare la realizzazione di collegamenti elettrici errati già durante la fase di disegno. La funzione di numerazione automatica dei componenti consente l'utilizzo di una numerazione selettiva e dipendente dalla pagina. Per l'aggiornamento degli attributi dei componenti è prevista un'apposita funzione che consente di importare automaticamente le informazioni da un database esterno semplificando la stesura di ordini, preventivi ecc. CadLogix consente di esportare la lista delle connessioni in differenti formati per l'impiego con altri software quali ad esempio quelli per il disegno dei PCB.

SIMBOLI E COMPONENTI

Selezione e piazzamento nello schema di simboli e componenti tramite una semplice operazione di drag-and-drop oppure tramite una finestra di dialogo che presenta i simboli e i componenti organizzati per categorie. Sono incluse diverse librerie con oltre 20000 simboli e componenti di cui oltre 7000 sono già predisposti per la simulazione. Oltre quelle fornite è possibile realizzare delle proprie

librerie utilizzando gli strumenti messi a disposizione dal software. Si possono realizzare simboli dinamici. I simboli dinamici hanno la proprietà di poter cambiare forma. Ad esempio, il simbolo di un display a sette segmenti può essere realizzato in modo che sia possibile, una volta posizionato il simbolo nello schema, selezionare il numero visualizzato sul display.

PROGRAMMABILE

È possibile estendere le funzioni del programma mediante il linguaggio di programmazione delle macro. Ad esempio è possibile creare una procedura per importare un disegno memorizzato in un formato per il quale non è fornito un appropriato filtro d'importazione oppure si può creare un nuovo filtro per l'esportazione della netlist.
Il linguaggio di script deriva dal linguaggio C/C++ dal quale eredita la grammatica, gli operatori, gli elementi di punteggiatura, le parole riservate ecc. Questo consente a chi già conosce il linguaggio C/C++ di programmare direttamente senza dover apprendere un nuovo linguaggio di programmazione. Non è richiesto alcun sistema di sviluppo esterno. Tutto ciò che serve per la programmazione delle macro è già incluso e totalmente integrato nell'ambiente di sviluppo di CadLogix.

SIMULAZIONE ANALOGICA, DIGITALE E MISTA

CadLogix include un potente simulatore analogico in standard SPICE e un veloce simulatore di circuiti digitali consentendo di simulare circuiti con solo componenti analogici, circuiti misti con componenti sia analogici che digitali e circuiti con solo componenti digitali. Il simulatore lavora direttamente sullo schema (sia a foglio singolo che suddiviso in più pagine o in più documenti). È sufficiente disegnare lo schema e selezionare il tipo di analisi per vedere immediatamente i risultati.
Ogni segnale presente nel circuito può essere riportato in un diagramma semplicemente posizionando una sonda di misura sul nodo da analizzare. Al termine della simulazione vengono visualizzate le forme d'onda di tutti i segnali indicati dalle sonde di misura presenti nello schema. È possibile vedere tutte le forme d'onda contemporaneamente oppure due o soltanto una. Tramite i cursori si possono effettuare misurazioni e marcare punti sul diagramma. Sono disponibili funzioni di zoom per visualizzare tutto o soltanto una parte del diagramma, per spostare il cursore sul minimo o il massimo ecc. Non è necessario essere degli esperti di SPICE per eseguire le simulazioni. La sintassi è gestita direttamente dal software. Naturalmente per gli utenti esperti è possibile specificare direttamente le istruzioni ed anche simulare direttamente circuiti descritti solamente tramite una netlist.
Il simulatore analogico accetta tutti i modelli sviluppati in puro standard SPICE. La maggior parte dei produttori di componenti sviluppano modelli SPICE per una totale compatibilità con i simulatori analogici. La libreria SIMDEVICES fornita include oltre 7000 componenti completi di tutti i parametri richiesti per la simulazione. La libreria include componenti passivi (resistori, induttori e condensatori ecc.), semiconduttori (diodi, transistors, FET, MOSFET ecc.), generatori analogici e digitali, amplificatori operazionali, circuiti logici, switches, strumenti di misura ecc.

Simulazione interattiva

Il simulatore consente di seguire il funzionamento dei circuiti in un modo totalmente interattivo e visuale. Durante la simulazione è possibile modificare lo stato degli ingressi (ad esempio cambiando la posizione di un interruttore, di un potenziometro o di un pulsante) ed osservare immediatamente la risposta del circuito.

Creazione di nuovi modelli

L'ambiente di sviluppo integrato in CadLogix consente di realizzare velocemente nuovi modelli di simulazione. Non è richiesto alcun sistema di sviluppo esterno. Tutto ciò che serve per la programmazione dei modelli è già incluso e totalmente integrato nell'ambiente di sviluppo di CadLogix.

RIFERIMENTI

Ulteriori informazioni su CadLogix possono essere trovate sul sito www.alterlogix.com dove è disponibile per il download una versione di prova del software.

Scheda di richiesta abbonamento

FE - 234

Si, desidero abbonarmi a FareELETTRONICA a partire dal primo numero raggiungibile

Cognome ______________________ Nome ______________________

Azienda ______________________

Via ______________________ CAP ________ Città ______________________ Prov. ________

Tel. ______________________ Fax ______________________ email ______________________

Abbonamento: *(barrare la casella prescelta)*

❑ **Standard:** Mi abbono a Fare Elettronica per un anno (11 uscite) a soli € 39,00 anziché € 51,00

❑ **Rinnovo:** Sono già abbonato ed intendo rinnovare il mio abbonamento in scadenza. Fare Elettronica per un anno (11 uscite) a soli € 39,00 anziché € 51,00, il mio codice abbonamento è ______________________

❑ **Regalo:** Regalo ad un amico Fare Elettronica per un anno (11 uscite) a soli € 35,00 anziché € 51,00
Riservato agli abbonati, il mio codice abbonamento è ______________________

❑ **Scuole:** Cinque abbonamenti a Fare Elettronica per un anno (11 uscite) a soli € 156,00 anziché € 195,00
Riservato a Scuole ed Università

Pagherò con: *(barrare la casella prescelta)*

❑ **Bollettino postale** Utilizzare il **C/C N. 22790232** intestato ad **Inware srl**,
indicando nella causale **"Abbonamento a Fare Elettronica"**

❑ **Bonifico bancario** Appoggiarlo su: **Poste Italiane**
CIN: Z - ABI: 07601 - CAB: 01600 - C/C: 000022790232 intestato ad **Inware srl**

❑ **Carta di credito** VISA MasterCard
Titolare: ______________________
Numero: ______________________ Scadenza: ____ / ____

Per completare l'attivazione dell'abbonamento, prego comunicare gli estremi (data e modalità prescelta) dell'avvenuto pagamento via telefono al numero (+39) 02.66504794 o via fax al numero (+39) 02.66508225

Firma ______________________

Abbonati subito!

- Compila il coupon e invialo via fax al numero 02.66508225
- Abbonati on-line: www.farelettronica.com
- Spedisci questo coupon in una busta chiusa a INWARE Edizioni
  Via Cadorna, 27/31 - 20032 Cormano (MI)
- Chiamaci al numero 02.66504794

INWARE EDIZIONI